MANUAL TÉCNICO

MANUAL TECNICO

Introducción

El Objetivo de este manual es poder ayudar a entender la estructura y el funcionamiento de la aplicación de tal manera que los programadores puedan ajustarlo a cambios deseados que puedan hacerse en un futuro.

Estructura del Gestor de Cobranzas.

El gestor de cobranzas está diseñado para tres perfiles que son: Administrador, ejecutivo y el usuario, el cual estará disponible para cuatro diferentes ambientes o Módulos que son: Form Reports, Sitio Web, Interfaz Móvil y el Cubo.

Modulo Form: consta de varias capas como: Capa Business, Data, Info las mismas que serán reutilizadas para los otros ambientes, por ese motivo se elaboró el aplicativo en forma estructurada para mostrar la interoperabilidad del aplicativo.

- ✓ **Cobranza. Configuración:** Establece la configuración con la base
- ✓ **CobranzaInfo**: Contiene todas las estructuras de las clases que vamos a usar
- ✓ **Cobranza Data**: capa que se conecta con la base para enlazar los datos por ejecuta las llamadas de los Sp. Insert, delete.
- ✓ **CobranzaBussiness**: Reglas del negocio va a contener todos los métodos, capa con la cual interactúan los programadores
- ✓ **Cobranza Schema3**: Valida los datos enviados
- ✓ **Cobranza.Winform:** Capa para mostrar los reportes del cubo
- ✓ Cada capa cumple con una funcionalidad especial dentro del proyecto la mima que a su vez contienen varias clases las cuales estaremos explicando en las siguientes líneas:

Modulo Web: Proyecto que servirá para mostrar los reportes desde el la Web; para el cual reutilizaremos el código del Ambiente Form del Visual Studio 2008; permitiéndonos mostrar la interoperabilidad del aplicativo

Modulo Movil.- Proyecto que servirá para mostrar los reportes desde el emulador del dispositivo móvil de Visual Studio 2008, que para llevar acabo dicho objetivo se instaló el Software de Microsoft Windows Mobile 6 Profesional.

Modulo: Cubo Este modulo permitirá mostrar la información a través de un Datamark para hacer análisis de la gestión de cobranzas.

Diagrama Entidad Relación del Cubo de Cobranzas

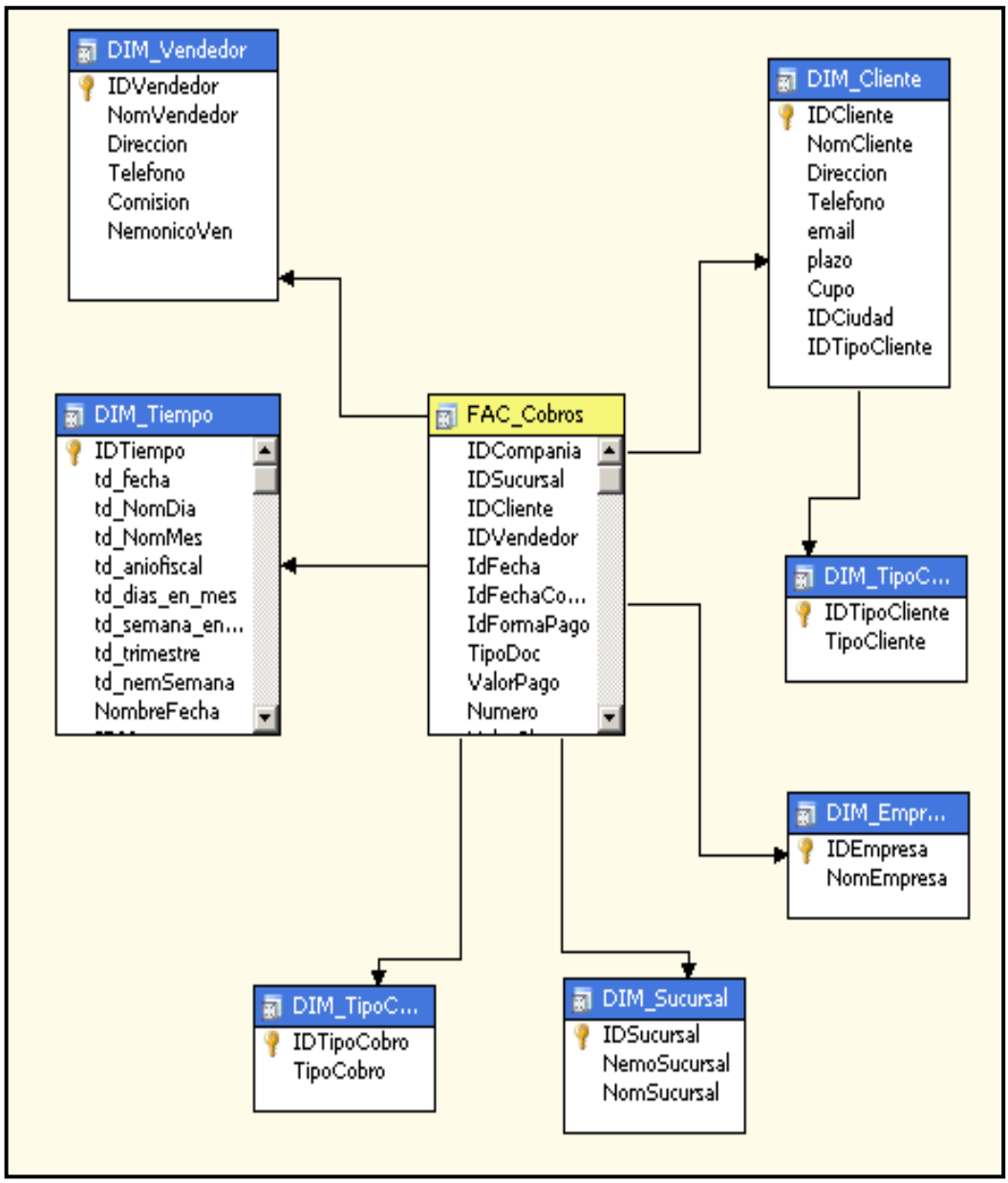


Tabla 0: Diagrama Entidad Relación del Cubo de Cobranzas

Detalle de Tablas

Las estructuras creadas para el desarrollo del presente proyectos son:

- ✓ **Tabla tb_Menu**

La finalidad de la creación de esta estructura es registrar los tipos de perfiles que pueden ingresar al aplicativo. A continuación se describirán cada uno de los campos de la estructura de la tabla Menú:

Nombre	Tipo Dato	Descripción
IDMenu	Int	Código del tipo de perfil
NombreMenu	**nchar(50)**	**Descripción del nombre del menú al cual se tiene privilegio**
Administrador	nchar(1)	Registra el tipo de perfil que está ingresando. En este caso el Administrador
Ejecutivo	nchar(1)	Registra el tipo de perfil que está ingresando. En este caso Ejecutivo.
Usuario	nchar(1)	Registra el tipo de perfil que está ingresando. En este caso Usuario Operador.
NameFrom	nchar(100)	Almacena el nombre de los formularios con los que interactúa el aplicativo.

Tabla 1: Tabla tb_Menu

- ✓ **Tabla tb_Grupo_Rpt**

La finalidad de la creación de esta estructura es registrar el grupo de reporte al que se está accesando. A continuación se describirán cada uno de los campos de la estructura:

Nombre	Tipo Dato	Descripción
IDGrupo	Int	Registra el código del grupo de reportes al cual va a ingresar.
Descripción	nchar(50)	Descripción el código del grupo de reportes.

Tabla 2: Tabla tb_Grupo_Rpt

- ✓ **Tabla tb_Reportes**

La finalidad de la creación de esta estructura es para registrar los reportes que interviene en el aplicativo. A continuación se describirán cada uno de los campos de la estructura.

Nombre	Tipo Dato	Descripción
IdReport	Int	Código del Reporte
NombreRpt	nchar(100)	Registra el nombre del reporte.
VistoXAdministrador	nchar(1)	Representa el estado del reporte que será visto por el administrador.
VistoxEjecutivo	nchar(1)	Representa el estado del reporte que será visto por el ejecutivo.
VistoxUser	nchar(1)	Representa el estado del reporte que será visto por el usuario.
NombreFuenteRpt	nchar(100)	Registra el nombre con que se crea el reporte en el formulario.
IdGrupo	Int	Código del grupo al que pertenece el reporte.
Estado	nchar(1)	Registra el estado del reporte “A” Activo, “I” Inactivo.

Tabla 3: Tabla tb_Reportes

- ✓ **Tabla tb_usuario**

La finalidad de la creación de esta estructura es para registrar al usuario que interactuará con el aplicativo. A continuación se describirán cada uno de los campos de la estructura.

Nombre	Tipo Dato	Descripción
tb_ldusuario	nchar(15)	Registra el código del usuario que está ingresando.
tb_nombre	nchar(30)	Registra el nombre representativo del Id_User.
tb_Administrador	nchar(1)	Registra si el tipo de perfil que tendrá el usuario es el de Administrador.
tb_ejecutivo	nchar(1)	Registra si el tipo de perfil que tendrá el usuario es el de Ejecutivo.
tb_user	nchar(1)	Registra el tipo de perfil que tendrá el usuario es el de usuario operador.
tb_password	nchar(20)	Registra la clave que tendrá el usuario al ingresar.

Tabla 4: Tabla tb_usuario

- ✓ **Tabla TbRe_FileXmlxTabla**

La finalidad de la creación de esta estructura es para registrar las relaciones entre los campos de las tablas de la base con las tablas proporcionadas por el cliente.

A continuación se describirán cada uno de los campos de la estructura.

Nombre	Tipo Dato	Descripción
IdTabla	char(50)	Registra el nombre de la tabla a relacionarse desde la base del cliente.
IdXml	char(50)	Registra el nombre de la tabla del XML a relacionarse desde la base del aplicativo.
IdCampoTabla	char(50)	Nombre del campo a relacionarse – base del cliente.
IdCampoXml	char(50)	Nombre del campo a relacionarse – base del aplicativo.
Enable	char(20)	Registra el estado del campo"1" si es activado y "0" si no lo está.

Tabla 5: Tabla TbRe_FileXmlxTabla

- ✓ **Tabla tbIN_Categorias**

La finalidad de la creación de esta estructura es para registrar las categorías de los productos que tienen las compañías.

A continuación se describirán cada uno de los campos de la estructura.

Nombre	Tipo Dato	Descripción
ca_compañia	Int	Registra el código de la compañía.
ca_IdCategoria	Int	Registra el código de la categoría.
ca_Codigo	nchar(10)	Registra el código
ca_Nombre	Varchar(50)	Registra el nombre de las categorías del producto.
ca_IdPadre	Int	
ca_fullPath	Varchar(1000)	

Tabla 6: Tabla TbIN_Categorias

- ✓ **Tabla tb_empresa**

La finalidad de la creación de esta estructura es para registrar los datos de la empresa

A continuación se describirán cada uno de los campos de la estructura.

Nombre	Tipo Dato	Descripción
em_IdEmpresa	Int	Código de la Empresa
em_nombre	nvarchar(50)	Registra el nombre de la empresa.
em_gerente	nvarchar(50)	Registra el nombre del gerente de la empresa.
em_ruc	nvarchar(13)	Registra el Ruc.
em_telefonos	varchar(100)	Guarda los teléfonos.
em_fax	char(20)	Guarda el número del fax de la empresa.
em_direccion	varchar(200)	Registra la dirección de la empresa.
em_imagen	image	Guarda la imagen de la empresa
em_logo	image	Guarda el logo de la empresa.

Tabla 7: Tabla tb_empresa

- ✓ **Tabla tbIN_Marca**

La finalidad de la creación de esta estructura es para registrar la marca de los productos con los que cuneta la compañía. A continuación se describirán cada uno de los campos de la estructura.

Nombre	Tipo Dato	Descripción
ma_Idcompania	Int	Código de la compañía.
ma_clave	Int	Registra el código clave de la marca.
ma_descripcion	varchar(100)	Nombre de la marca
ma_estado	char(1)	Registra el estado de la marca "A" si esta activo y por consiguiente "I" si está inactivo.

Tabla 8: Tabla tbIN_Marca

- ✓ **Tabla tbSucursal**

La finalidad de la creación de esta estructura es para registrar los datos de las sucursales que tienen algunas empresas.

A continuación se describirán cada uno de los campos de la estructura.

Nombre	Tipo Dato	Descripción
su_Idcompania	Int	Código de la compañía.
su_IdSucursal	Int	Describe el IDSucursal.
su_Codigo	nvarchar(10)	Representa el código de la sucursal.
su_Nemonico	char(4)	Describe en tres letras la sucursal.
su_nombre	nvarchar(50)	Nombre de la Sucursal
su_ruc	nvarchar(15)	Guarda el Ruc de cada sucursal
su_responsable	nvarchar(50)	Registra el nombre del gerente para cada sucursal.
su_direccion	nvarchar(80)	Describe la dirección
su_telefono	nvarchar(50)	Guarda los teléfonos
su_IdCiudad	Int	Guarda el cd de la ciudad a la que pertenece la sucursal.
su_estado	char(1)	Guarda el estado de la sucursal "A" si esta activada.

Tabla 9: Tabla tbSucursal

- ✓ **Tabla tbCC_Cliente**

La finalidad de la creación de esta estructura es para registrar los datos de los clientes con los que cuentan las empresas. A continuación se describirán cada uno de los campos de la estructura.

Nombre	Tipo Dato	Descripción
cl_Idcompania	Int	Código de la compañía.
cl_IdCliente	numeric(18,0)	Codigo del cliente.
cl_IdSucursal	Int	Codigo de sucursal.
cl_cedula_ruc_Pass	nvarchar(15)	Registra la cedula o el Ruc del cliente.
ca_Nombre	nvarchar(100)	Nombre del Cliente
cl_RazonSocial	nvarchar(100)	Registra el nombre de la Razón Social.
cl_direccion	nvarchar(130)	Guarda la dirección del cliente.
cl_telefono	nvarchar(100)	Registra los teléfonos
cl_email	nvarchar(50)	Guarda la dirección del correo electrónico en caso de tenerlo.
cl_tipo_cliente	Tinyint	Describe el tipo de cliente con que cuenta la empresa.
cl_credito	nvarchar(1)	Registra si el cliente tendrá crédito o no.
cl_plazo	Smallint	Registra los días de crédito que tendrá el cliente.
cl_estado	nvarchar(1)	Registra si el cliente está Activo o Inactivo.
cl_observacion	varchar(130)	Guarda datos adicionales del cliente.
cl_fecha	datetime	Guarda la fecha de inscripción del cliente.
cl_IdCiudad	Int	Registra el código de la ciudad del cliente.
cl_Idzona	Int	Guarda la zona a la que pertenece el cliente.
cl_Cupo	float	Cupo de crédito que tendrá el cliente.
cl_fax	varchar(50)	Registra el fax del cliente en caso de tenerlo.

Tabla 10: Tabla tbCC_Cliente

- ✓ **Tabla tbCC_CobrosxCliente**

La finalidad de la creación de esta estructura es para registrar los cobros realizados a los clientes. A continuación se describirán cada uno de los campos de la estructura.

Nombre	Tipo Dato	Descripción
cr_Idcompania	Int	Código de la compañía.
cr_Idsucursal	Int	Codigo de sucursal.
cr_numero	numeric (18,0)	Número del documento del cobro.
cr_Idcliente	numeric(10,0)	Codigo del cliente.
cr_IdFormaPago	char(3)	Registra el código de tres letras para la forma de pago efec,chv.
cr_valor	Money	Representa el valor del pago.
cr_fecha	date	Registra la fecha actual.
cr_fechaCobro	datetime	Fecha en que se realizo el cobro.
cr_fechaEdicion	datetime	Fecha en que se emitió el documento para crear el cobro.
cr_observacion	nvarchar(100)	Descripción del cobro.
cr_Banco	Smallint	Representa que el pago es en cheque.
cr_cuenta	nvarchar(100)	Registra el número de la cuenta bancaria.
cr_NumCheque	nvarchar(100)	Describe el # del cheque.
cr_Tarjeta	Smallint	Documento para registrar el pago.
cr_num Tarj	nvarchar (100)	Describe el # de tarjeta para efectuar el pago.
cr_saldo	Money	Representa el saldo de Cta. x Pagar.
cr_fechaReg	datetime	Fecha de registro del pago.

Tabla 11: Tabla tbCC_CobrosxCliente

- ✓ **Tabla tbCC_CobrosxCliente_Det**

La finalidad de la creación de esta estructura es para registrar el detalle de la cabecera de las cuentas por cobrar.

Nombre	Tipo Dato	Descripción
dr_Idcompania	Int	Código de la compañía.
dr_Idsucursal	Tinyint	Código de sucursal.
dr_numero	numeric(18,0)	Número del documento.
dr_secuencial	Smallint	Secuencia del documento de pago.
dr_tipoDoc	nchar(1)	Tipo de documento del pago.
dr_Num_fact_Nota	numeric (18,0)	# de factura o nota de venta.
dr_ValorPago	Money	Valor de pago de factura
dr_Vendedor	Int	Código vendedor del cliente.

Tabla 12: Tabla tbCC_CobrosxCliente_Det

- ✓ **Tabla tbCC_NotaCredDeb**

La finalidad de la creación de esta estructura es para registrar los créditos o débitos que tienen los clientes a través de este tipo de documento.

A continuación se describirán cada uno de los campos de la estructura.

Nombre	Tipo Dato	Descripción
sc_Idcompania	Int	Código de la compañía.
sc_Idsucursal	Int	Código de sucursal.
sc_Idcliente	Int	Código del cliente.
sc_Idvendedor	Int	Código del vendedor.
sc_fecha	Datetime	Registra la fecha actual.
sc_serie_fac	char(7)	Registra la serie de la factura.
sc_factura	numeric(18,0)	Registra el # de la factura.
sc_fecha_emision	Datetime	Fecha en que es emitida N/Crédito Debito.
sc_IdTipo	nchar(10)	Código del tipo de Nota.
sc_IdMotivo	Int	Motivo por el cual se emite la nota.
sc_Idempleado	Int	Registra el código del empleado.
sc_observacion	varchar(1000)	
sc_subtotal	Money	Calculo del subtotal.
sc_descuento	Money	Calculo de los descuentos.
sc_iva	Money	Calculo del IVA.
sc_flete	Money	Calculo del flete.
sc_interes	Money	Calculo del interés.
sc_total	Money	Calcula el total.

Tabla 13: Tabla tbCC_NotaCredDeb

- ✓ **Tabla tbCC_NotaCredDeb_Det**

La finalidad de la creación de esta estructura es para registrar los detalles de la cabecera de la NotaCreditoDebito.

A continuación se describirán cada uno de los campos de la estructura.

Nombre	Tipo Dato	Descripción
sc_compania	Int	Nombre de compañía.
sc_sucursal	Int	Registra el código de la sucursal.
sc_secuencia	Smallint	Secuencia de la Nota C/D.
sc_producto	numeric(10,0)	Cód. del producto por el cual se emite la nota.
sc_cantidad	Float	Usuario que realizó la modificación del estado.
sc_pvp	Money	Precio del producto.
sc_por_des	Money	Descuentos del producto.
sc_descUni	Float	Descuento por unidades.
sc_desc	Money	Descuento total.
sc_precioFinal	Money	Precio final del producto.
sc_subtotal	Money	Calculo del subtotal.
sc_observacion	varchar(1000)	
sc_subtotal	Money	Calculo del subtotal.
sc_iva	Money	Calculo del IVA.
sc_total	Money	Total de Nota.

Tabla 14: Tabla tbCC_NotaCredDeb_Det

- ✓ **Tabla tbCC_TipNota**

La finalidad de la creación de esta estructura es para registrar el tipo de Nota que se aplicará al cliente.

A continuación se describirán cada uno de los campos de la estructura.

Nombre	Tipo Dato	Descripción
tn_Idcompania	Tinyint	Código de la compañía.
tn_IdTipoNota	Int	Código del tipo nota.
tn_Tipo	char(1)	Tipo de nota.
tn_descripcion	varchar(50)	Registra datos adicionales de la nota.
tn_estado	varchar(1)	Registra "A" si esta activa la nota.
tn_cuenta	varchar(20)	Registra la cuenta del cliente al que se le aplica la nota.
tn_interno	char(10)	Registra si la nota fue emitida por motivo interno de la Cia.
tn_PorRetencion	char(10)	Nota Emitida por retención.
tn_PorRetIva	char(1)	Nota emitida por retención Iva.
tn_PorRetFuente	char(1)	Nota emitida por retención Fuente.
tn_RestaAVentas	char(1)	Nota emitida por descuento en ventas.
tn_SePagaComisiones	char(4)	Nota emitida x pago de comisiones.

Tabla 15: Tabla tbCC_TipNota

- ✓ **Tabla tbCC_TipoCliente**

Estructura creada para registrar los tipos de clientes. A continuación se describirán cada uno de los campos de la estructura.

Nombre	Tipo Dato	Descripción
IDTipoCliente	Tinyint	Registra el código tipo de cliente.
tc_descripción	char(20)	Describe el tipo del cliente.

Tabla 16: Tabla tbCC_TipoCliente

- ✓ **Tabla tbCC_TipoformaCobro**

La finalidad de la creación de esta estructura es para guardar los datos realizados de cobros a clientes. A continuación se describirán cada uno de los campos de la estructura.

Nombre	Tipo Dato	Descripción
IDTipoCobro	char(3)	Código del tipo cobro.
tf_descripcion	nvarchar(30)	Descripción del tipo cobro.
tf_banco	char(1)	Registra “S” si el documento es un cheque, “N” en caso no serlo.
tf_DocAfecha	nvarchar(1)	Registra “S” si el Ch es a fecha “N” en caso de no serlo.
tf_interno	char(1)	Registra “S” si el cobro es interno de la Cía. Y “N” en caso de no serlo.
tf_estado	char(1)	Si el cobro está activo o no.
tf_nemonico	char(5)	Registra en 5 letras la forma de tipo de cobro.
tf_cobroDirecto	char(1)	Registra si el cobro fue directo al cliente o a través efectivo, ch.Vista
tf_cobroInDirecto	char(1)	Si el cobro fue efectuado con cheques posfechado.
tf_docXCobrar	char(1)	Registra el doc. para el cobro “S” o N”.
tf_Orden	Tinyint	Fecha de modificación del rol.
tf_nemonico2	char(10)	Registra 10 caracteres del doc. para forma de cobro

Tabla 17: Tabla tbCC_TipoformaCobro

- ✓ **Tabla tbTarjCredit**

La finalidad de la creación de esta estructura es para registrar una de las formas de pago y los datos que intervienen en ella.

A continuación se describirán cada uno de los campos de la estructura.

Nombre	Tipo Dato	Descripción
IdTarjeta	char(3)	Registra el IdTarjeta de tres letras.
tj_codigo	nvarchar(15)	Registra el código de tarjeta.
tj_descripción	nvarchar(50)	Describe el nombre de la tarjeta.
tj_estado	nvarchar(1)	Estado de la tarjeta. Estado "a" de activo y estado "I" de Inactivo.
tj_ctaCble	nvarchar(20)	Número de cuenta de la tarjeta del cliente.

Tabla 18: Tabla tbTarjCredit

- ✓ **Tabla tbIN_TipoProducto**

La finalidad de la creación de esta estructura es para registrar los tipos de productos con los que cuenta la empresa. A continuación se describirán cada uno de los campos de la estructura.

Nombre	Tipo Dato	Descripción
IDTipoProducto	Number	Código de la otorgación.
tp_descripción	varchar2(100)	Descripción de la otorgación.

Tabla 19: Tabla tbIN_TipoProducto

- ✓ **Tabla tbIN_Producto**

La finalidad de la creación de esta estructura es para registrar los productos disponibles para los clientes. A continuación se describirán cada uno de los campos de la estructura.

Nombre	Tipo Dato	Descripción
pr_IdCompania	Int	Código de la compañía.
pr_codigo	nvarchar(40)	Registra el código de los productos.
pr_TipoProduc	char(3)	Describe el tipo de producto.
pr_IdCategoria	Int	Registra el Id Categoría.
pr_descripcion	nvarchar(500)	Registra el nombre del producto.
pr_pvp	Money	Describe el precio de venta.
pr_pvp_alxMayor	Money	Fija el precio de venta al por mayor.
pr_stock	float	Cantidad disponible en el stock.
pr_pedidos	float	Registra los pedidos por producto.
pr_IdFabricante	Smallint	Registra el código del fabricante.
pr_Manejalva	nvarchar(1)	Informa si el producto paga o no IVA.
pr_estado	Tinyint	Estado del producto activo o inactivo.
pr_precio_minimo	Money	Registra el precio mínimo.
pr_IdProveedor	Smallint	Registra el código del proveedor.
pr_IdMarca	Int	Describe el código de la marca.
pr_fecha	Datetime	Registra la fecha de ingreso del producto.

Tabla 20: Tabla tbIN_Producto

- ✓ **Tabla tbFa_Factura**

La finalidad de la creación de esta estructura es para registrar todas las facturas emitidas a los clientes. A continuación se describirán cada uno de los campos de la estructura.

Nombre	Tipo Dato	Descripción
vt_IdEmpresa	Int	Código de la empresa que factura.
vt_IdSucursal	Int	Registra el código de la sucursal que factura.
vt_IdFactura	numeric(10,0)	Registra el Id de la factura.
vt_Factura	numeric(18,0)	Registra el código de la factura.
vt_IdVendedor	Int	Registra el código del vendedor.
vt_IdCliente	numeric(10,0)	Registra el código del cliente.
vt_fecha	Datetime	Registra la fecha de la factura.
vt_base_coniva	Money	Registra la base a la que se le calcula el IVA.
vt_base_siniva	Money	Registra la base que va sin IVA.
vt_subtotal	Money	Registra el subtotal.
vt_descuento	Money	Registra el descuento de la factura.
vt_iva	Money	Registra el IVA de la factura.
vt_total	Money	Registra el valor total de la factura.
vt_estado	nvarchar(1)	
vt_CreCon	char(3)	
vt_plazo	numeric(10,0)	Registra el plazo de la factura.
vt_fech_venc	datetime	Registra la fecha de vencimiento de la factura.
vt_saldo	Money	Registra el saldo de la factura.
vt_seguro	Money	Registra el seguro.
vt_flete	Money	Registra el flete de la factura.
vt_Observacion	varchar(200)	Se pone la observación sobre la factura.
vt_tipo	varchar(2)	
vt_usuario	char(20)	Registra el nombre del usuario que realiza la factura.

Tabla 21: Tabla tbFa_Factura

- ✓ **Tabla tbFa_Factura_Det**

La finalidad de la creación de esta estructura es para registrar todos los detalles de las facturas emitidas a los clientes. A continuación se describirán cada uno de los campos de la estructura.

Nombre	Tipo Dato	Descripción
vt_Idcompania	Tinyint	Registra el código de la compañía.
vt_IdSucursal	Int	Registra el código de la sucursal.
vt_IdFactura	numeric(10,0)	Registra el Id de la factura.
vt_secuencia	Smallint	Registra la secuencia de la factura.
vt_IdProducto	numeric(10,0)	Registra el código del producto.
vt_cantidad	Money	Registra la cantidad del producto.
vt_Pvp	Money	Registra el precio unitario del producto.
vt_DescUni	Money	Registra el descuento unitario del producto.
vt_DescTotal	Money	Registra el descuento total de la factura.
vt_PvpFinal	Money	Registra el precio total.
vt_Subtotal	Money	Registra el subtotal de la factura.
vt_ivaUni	Money	Registra el IVA de cada uno de los productos.
vt_IvaTotal	Money	Registra el total del IVA.
vt_total	Money	Registra el valor total de la factura.
vt_costo	Money	
vt_tieneIVA	char(1)	

Tabla 22: Tabla tbFa_Factura_Det

- ✓ **Tabla tbFa_Vendedor**

La finalidad de la creación de esta estructura es para registrar los datos del vendedor. A continuación se describirán cada uno de los campos de la estructura.

Nombre	Tipo Dato	Descripción
vd_Idcompania	Tinyint	Registra el código de la compañía.
vd_IdVendedor	Int	Registra el código del vendedor.
vd_IdSucursal	Tinyint	Registra el código de la sucursal.
vd_Codigo	nvarchar(10)	Registra el código de la factura.
vd_nombre	nvarchar(50)	Registra el nombre del vendedor.
vd_direccion	nvarchar(50)	Registra la dirección del vendedor.
vd_telefono	nvarchar(20)	Registra el teléfono del vendedor.
vd_fech_ing	Datetime	Registra la fecha de ingreso del vendedor.
vd_tipo	Smallint	Registra que tipo de vendedor es.
vd_comisión	Money	Registra el valor de la comisión del vendedor.
vd_estado	nvarchar(1)	Registra el estado del vendedor "A" activo e "I" está "Inactivo.
vd_nemonico	nvarchar(20)	

Tabla 23: Tabla tbFa_Vendedor

- ✓ **Tabla RptCC_CobrosxCliente**

La finalidad de la creación de esta estructura es para presentar los reportes con sus respectivas especificaciones. A continuación se describirán cada uno de los campos de la estructura.

Nombre	Tipo Dato	Descripción
cr_Idcompania	Int	Registra el código de la compañía.
cr_Idsucursal	Int	Registra el código de la sucursal.
cr_numero	numeric(18,0)	
cr_Idcliente	numeric(10,0)	Registra el código del cliente.
cr_IdFormaPago	char(3)	Escoge la forma de pago.
cr_valor	Money	Indica el valor pendiente por pagar.
cr_fecha	date	Indica la fecha de la factura.
cr_fechaCobro	Datetime	Indica la fecha de cobro de la factura.
cr_fechaEdicion	Datetime	
cr_observacion	nvarchar(100)	Pone alguna observación en la factura.
cr_Banco	Smallint	Indica el nombre del banco.
cr_cuenta	nvarchar(100)	Indica el # cuenta del banco.
cr_NumCheque	nvarchar(100)	Indica el # de cheque con que cancela la factura.
cr_Tarjeta	Smallint	Indica de qué banco es la tarjeta.
cr_numTarj	nvarchar(100)	Indica el número de tarjeta con que cancela.
cr_estado	nvarchar(1)	Indica el estado.
cr_saldo	Money	Indica el saldo del cliente.
cr_fechaReg	Datetime	Registra la fecha del pago.
cr_IdFecha	Int	
cr_IdFechaCobro	Int	Indica la fecha del cobro.
PSucursal	char(20)	Indica de qué sucursal es el cliente.
PCliente	char(20)	Indica el nombre del cliente.
PVendedor	char(20)	Indica cual es el vendedor.
PDesde	datetime	Indica la fecha del reporte.
PHasta	datetime	Indica hasta que fecha desea el reporte.
PTipoCobro	char(20)	Indica que tipo de cobro se realizó.
PTipoRpt	char(20)	Indica el tipo de reporte que desea.
cr_plazo	Int	Indica el plazo de pago de la factura.

Tabla 24: Tabla RptCC_CobrosxCliente

- ✓ **Tabla RptCC_CobrosxProducto**

Nombre	Tipo Dato	Descripción
cr_Idcompania	Int	Registra el código de la compañía.
cr_Idsucursal	Int	Registra el código de la sucursal.
cr_numero	numeric(18,0)	
cr_Idcliente	numeric(10,0)	Registra el código del cliente.
cr_IdFormaPago	char(3)	Escoge la forma de pago.
cr_valor	Money	Indica el valor pendiente por pagar.
cr_fecha	date	Indica la fecha de la factura.
cr_fechaCobro	Datetime	Indica la fecha de cobro de la factura.
cr_fechaEdicion	Datetime	
cr_observacion	nvarchar(100)	Pone alguna observación en la factura.
cr_Banco	Smallint	Indica el nombre del banco.
cr_cuenta	nvarchar(100)	Indica el # cuenta del banco.
cr_NumCheque	nvarchar(100)	Indica el # de cheque con que cancela la factura.
cr_Tarjeta	Smallint	Indica de qué banco es la tarjeta.
cr_numTarj	nvarchar(100)	Indica el número de tarjeta con que cancela.
dr_secuencial	Smallint	Indica la secuencia.
dr_tipoDoc	nchar(1)	Indica el tipo de documento.
dr_Num_fact_Nota	numeric(18,0)	
dr_ValorPago	Money	Indica el valor a pagar.
vt_IdFactura	numeric(10,0)	Indica el código de la factura.
vt_Factura	numeric(18,0)	Indica el número de factura.
vt_fecha	datetime	Indica la fecha de la factura.
vt_CreCon	char(3)	
vt_plazo	numeric(10,0)	Indica el plazo para cobrar.
vt_fech_venc	datetime	Indica la fecha de vencimiento.
vt_IdProducto	numeric(10,0)	Indica el código del producto.
vt_cantidad	Money	Indica el valor del producto.
vt_Subtotal	Money	Indica el subtotal del producto.

Tabla 25: Tabla RptCC_CobrosxProducto

- ✓ **Tabla RptCC_CobrosxCliente_Det**

Nombre	Tipo Dato	Descripción
dr_Idcompania	Int	Registra el código de la compañía.
dr_Idsucursal	Tinyint	Registra el código de la sucursal.
dr_numero	numeric(18,0)	
dr_secuencial	Smallint	Indica la secuencia.
dr_tipoDoc	nchar(1)	Indica el tipo de documento.
dr_Num_fact_Nota	numeric(18,0)	
dr_ValorPago	Money	Indica el valor a pagar.
dr_Vendedor	Int	Indica el nombre del vendedor.
Id_Vendedor	varchar(15)	Registra el código del vendedor.

Tabla 26: Tabla RptCC_CobrosxCliente_Det

Diagrama Casos de Uso

El Diagrama de Casos de Uso es la técnica por la cual vamos representar la funcionalidad que ofrece el Cubo de Información de Gestión de Cobranzas en lo que se refiere a su interacción con el usuario para generar los respectivos reportes utilizados por el operador para gestionar las Cobranzas.

Para un mejor entendimiento hemos clasificado estos casos de uso por niveles, el nivel 0 representa un esquema básico del diagrama y el nivel 1 es donde detallaremos más acerca de cada uno de los procesos así como también describiremos los actores que intervienen en estos escenarios:

Definición de actores

ACTOR 01	Usuario - Gestor de Cobranzas
Descripción	Este actor representa al usuario que manipula el sistema
Comentarios	No es necesario de un usuario experto para utilizarlo
Simbología	ACTOR 01

Tabla27: Descripción Actor 1

ACTOR 02: Es la Base de Datos que estará usando el programa.

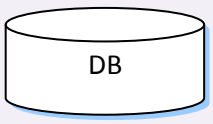
ACTOR 02	Base de Datos
Descripción	Almacenamiento de la información necesaria para el Cubo.
Comentarios	Se encuentra la información proporcionada x el cliente
Simbología	 ACTOR0

Tabla28: Descripción Actor 2

Caso de Uso del Nivel 0

Nivel 0:

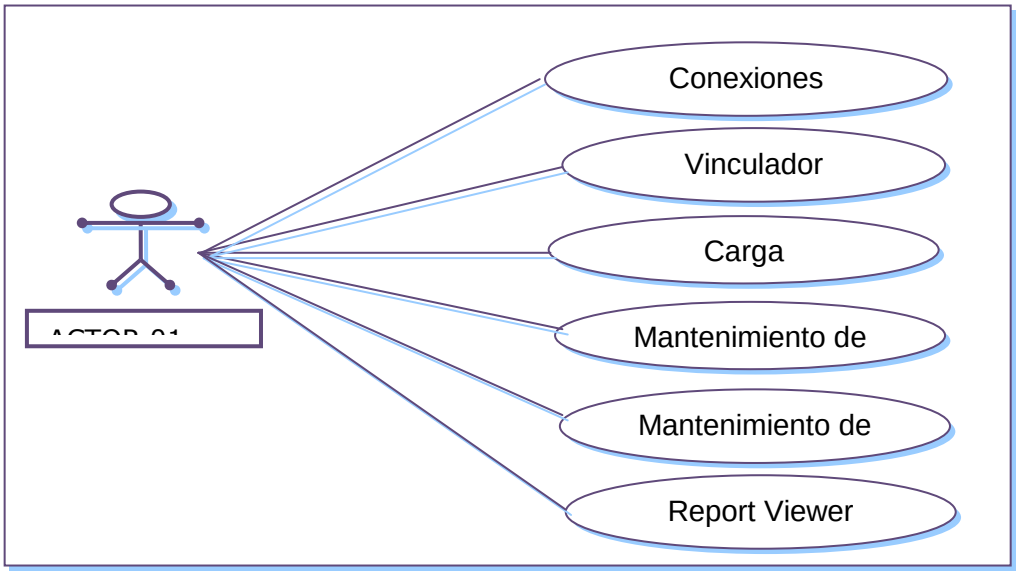


Figura 46: Caso de Uso: Ingreso al aplicativo

Caso de Uso	**Ingreso del Gestor**
Nombre:	Uso del Gestor de Cobranzas
Actores:	ACTOR 01
Precondiciones:	1.- Estar registrado o Autorizado 2.- Debe existir por lo menos una base de datos del cliente
Función:	1.- Iniciar sesión: Usuario + password. 2.- Realizar la Conexión con la B/D del cliente para vincular las tablas, validarlas con los esquemas y migrarlas para realizar la carga y poder mostrarla en los reportes.
Comentarios	El Administrador, ejecutivo y gestor ingresan de la misma forma.

Tabla29: Descripción caso de uso Ingreso

Nivel 1:

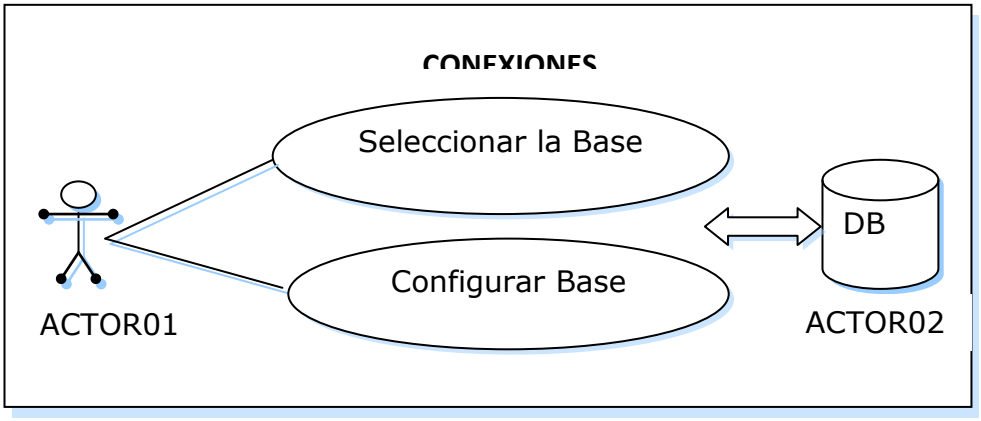


Figura 47: Caso de uso - Conexión

ID Caso de Uso	CONEXIONES
Nombre:	Conexión
Actores:	ACTOR 01 - ACTOR 02
Precondiciones:	Existir Por lo menos 1B/D del cliente.
Flujo de Datos:	1.- Seleccionar la B/D del Cliente a la cual deseamos conectarnos. 2.- Si es Access click en siguiente. 3.- Click Examinar y escoger la ruta de B/D, 4.- si es Microsoft SQL Server 2003 click Botón Configurar B/D 5.- Mensaje "Conexión OK" y Click en Botón Aceptar.
Comentarios	Si la conexión no es exitosa, pedirá que se conecte nuevamente.

Tabla 30: Descripción Caso de Uso – Conexiones

Nivel 1:

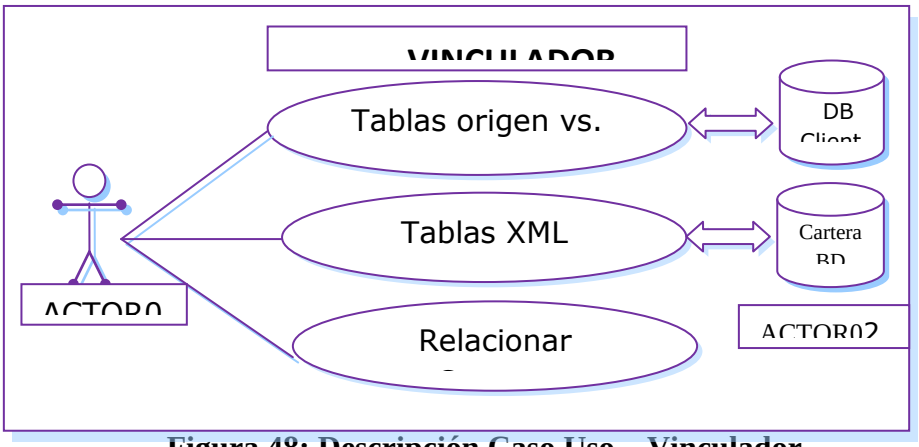


Figura 48: Descripción Caso Uso – Vinculador

Caso de Uso	Vinculador
Nombre:	Vinculador
Actores:	ACTOR 01 - ACTOR02
Precondiciones:	2.- Exista por lo menos 1 B/D Cliente
Flujo de Trabajo	1.- Seleccionar tabas del cliente vs los XML .para establecer la vinculación 2.- Seleccionar campos de las tablas a relacionar 3.- Verificar que se activa el estado del check box para que se establezca la relación entre campos
Comentarios	CarteraBD es la B/D del Cubo de Información y XML son los esquemas para comparar y validar

Tabla31: Descripción Caso de Uso – Vinculador

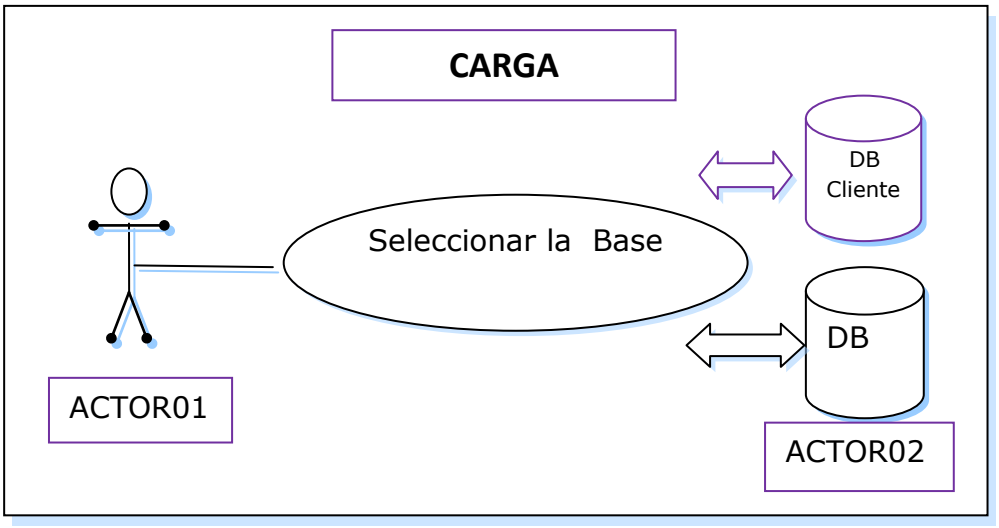


Figura 49: Carga

Caso de Uso	Carga
Nombre:	CARGA
Actores:	ACTOR 01 - ACTOR02
Precondiciones:	1.- Exista por lo menos una Vinculación entre tablas
Flujo de Trabajo	1.- Activar la casilla (check box) de las tablas que se desean cargar 2.- Click en botón Validación y carga.
Comentarios	Los campos obligatorios deben estar relacionados para que la carga se exitosa

Tabla32: Descripción Caso de Uso Carga

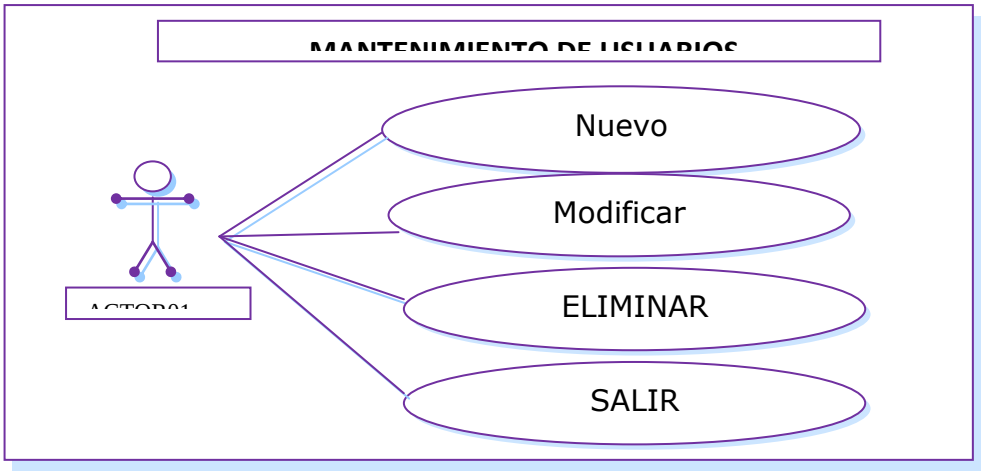


Figura50: Descripción Caso Uso – Mantenimiento de Usuario

Caso de Uso	Mantenimiento de Usuarios
Nombre:	MANTENIMIENTO USUARIOS
Actores:	ACTOR 01
Precondiciones:	1.- Exista por lo menos un usuario
Flujo de Trabajo	1.- Permitir crear nuevos usuarios, modificarlos y eliminarlos 2.- Click Salir, regresa pantalla principal.
Comentarios	ninguno

Tabla33: Descripción Caso de Uso Mantenimiento Usuario

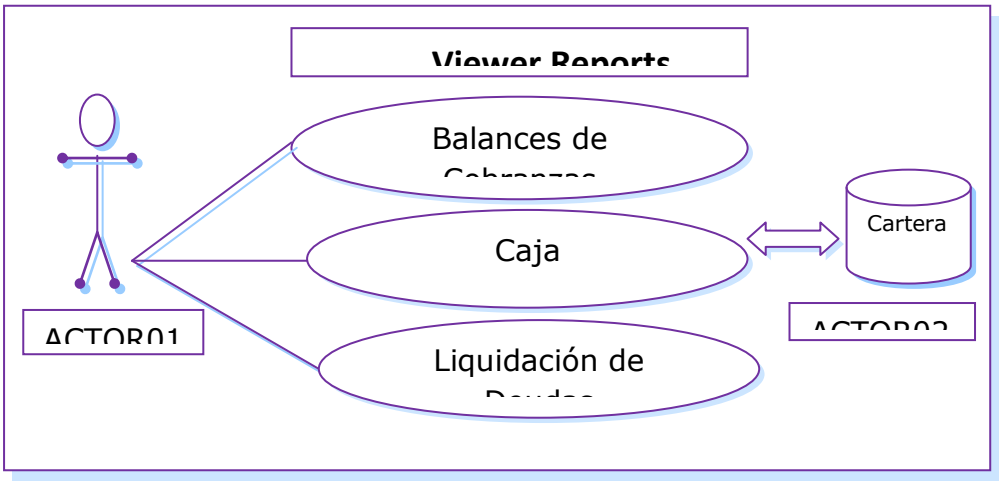


Figura 51: Viewer Reports

Caso de Uso	Viewer Reports
Nombre:	Viewer Reports
Actores:	ACTOR 02
Precondiciones:	1.- Existan los reportes
Flujo de Trabajo	1.- Hacer Click en Reportes 2.- Seleccionar cual de las carpetas se desea monitorear los reportes
Comentarios	Doble Click para mostrar toda la pantalla

Tabla34: Descripción Caso de Uso - Viewer Reports

Nivel 2:

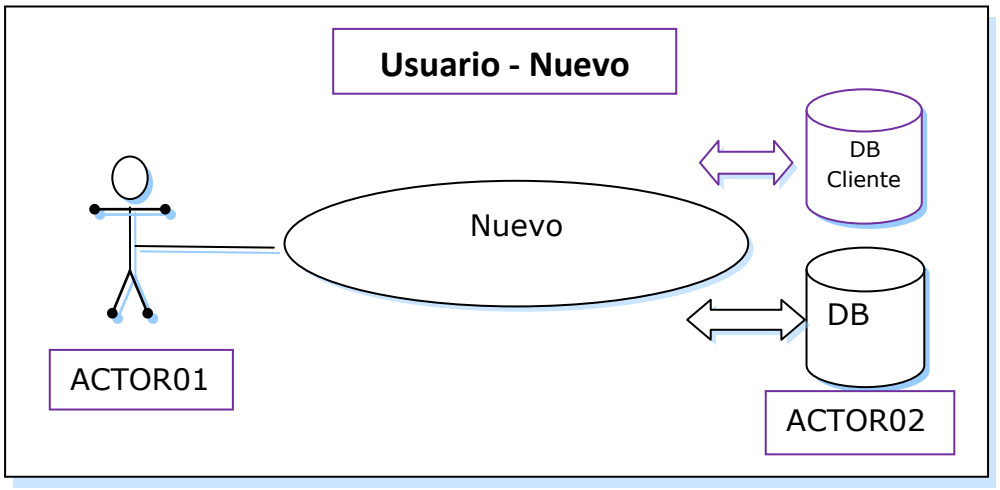


Figura 52: Descripción Mantenimiento Usuario – Nuevo

Caso de Uso	Mantenimiento Usuario - Nuevo
Nombre:	Usuario – Nuevo
Actores:	ACTOR 01 - ACTOR02
Precondiciones:	1.- Exista por lo menos un reporte creado
Flujo de Trabajo	1.- Asignar Login a Usuario Nuevo 2.- Ingresar el Alias del usuario. 3.- Ingresar el Password al usuario
Comentarios	Ninguna

Tabla35: Descripción Caso de Uso Mantenimiento Usuario

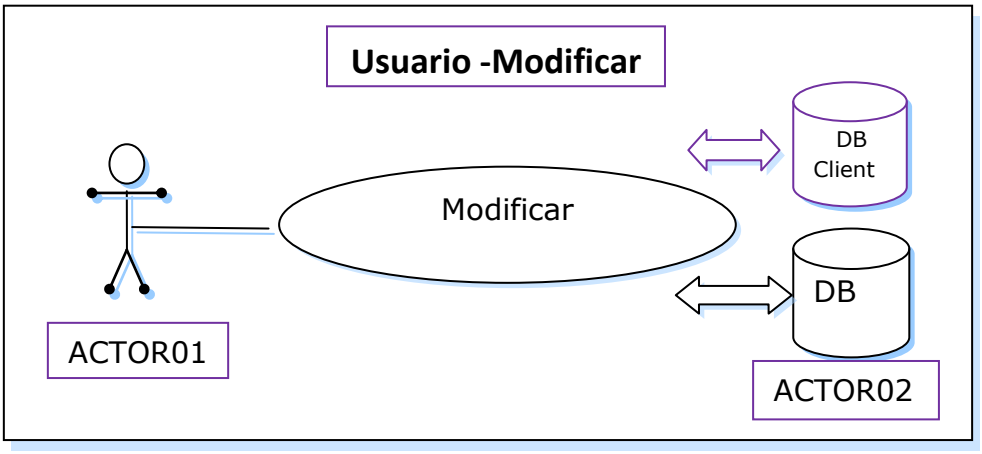


Figura53: Mantenimiento Usuario – Modificar

Caso de Uso	**Mantenimiento Usuario - Modificar**
Nombre:	Usuario - Modificar
Actores:	ACTOR 01 - ACTOR02
Precondiciones:	1.- Exista por lo menos un usuario creado
Flujo de Trabajo	1.- Cambiar Password Actual por el nuevo si ese fuera el caso.
Comentarios	Ninguna

Tabla36: Descripción Caso de Uso Mantenimiento Modificar

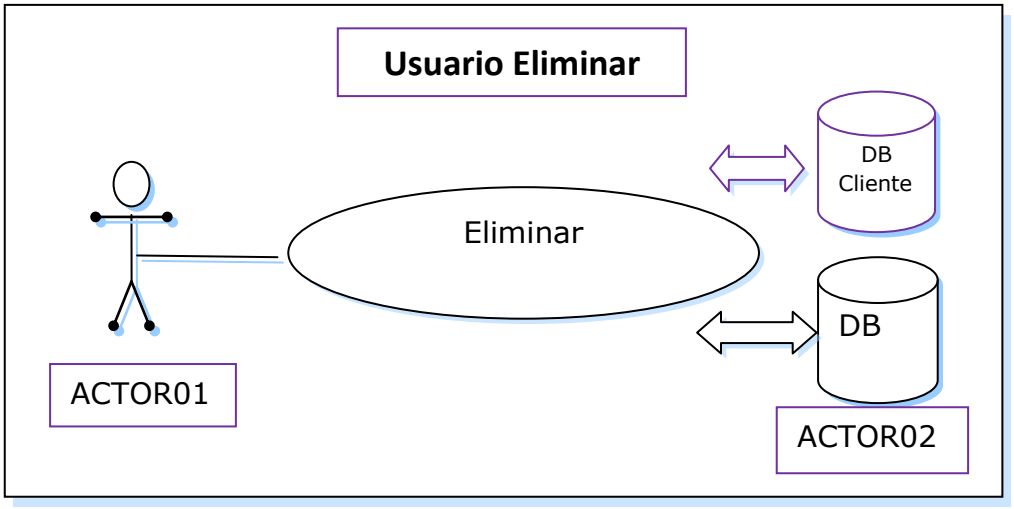


Figura 54: Descripción Caso Uso – Mantenimiento – Eliminar

Caso de Uso	Mantenimiento de Usuarios - Eliminar
Nombre:	ELIMINAR
Actores:	ACTOR 01 - ACTOR02
Precondiciones:	1.- Exista por lo menos un usuario
Flujo de Trabajo	1.- Borrar usuario
Comentarios	Ninguno

Tabla37: Descripción Caso de Uso Mantenimiento Usuario – Eliminar

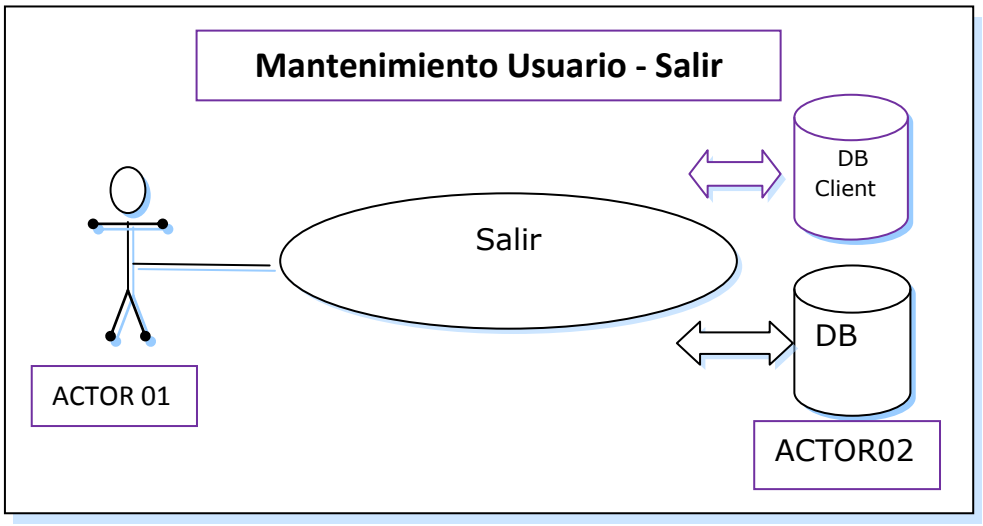


Figura 55: Mantenimiento Usuarios – Salir

Caso de Uso	Mantenimiento de Usuarios
Nombre:	SALIR
Actores:	ACTOR 01 -
Precondiciones:	1.- Haber navegado en el aplicativo
Flujo de Trabajo	1.- Hacer Click en botón Salir
Comentarios	Ninguno

Tabla38: Descripción Caso de Uso Mantenimiento Usuario – Salir

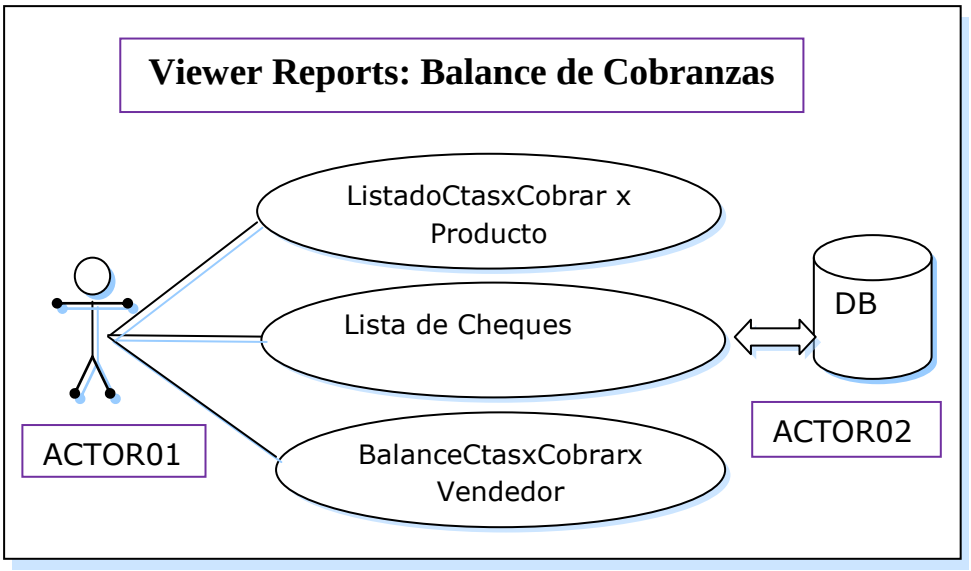


Figura 56: Viewer Reports – Balance de Cobranzas

Caso de Uso	Viewer Reports
Nombre:	Balance de Cobranzas
Actores:	ACTOR 01 – ACTOR02
Precondiciones:	1.- Existan Reportes
Flujo de Trabajo	1.- Hacer doble Click en el reporte 2.- Seleccionar los filtros deseados 3'- Click en Buscar
Comentarios	Es primordial seleccionar la Fecha

Tabla 39: Viewer Reports – Balance de Cobranza

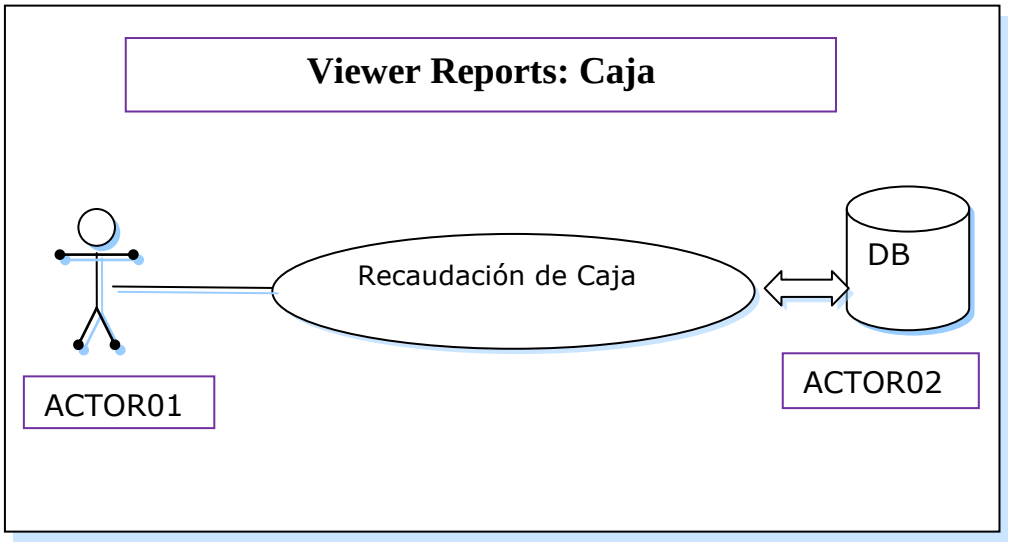


Figura 57: Viewer Reports – Recaudación Caja

Caso de Uso	Viewer Reports
Nombre:	Caja
Actores:	ACTOR 01 – ACTOR02
Precondiciones:	1.- Existan Reportes
Flujo de Trabajo	1.- Hacer doble Click en el reporte 2.- Seleccionar los filtros deseados 3'- Click en Buscar
Comentarios	Es primordial seleccionar la Fecha

Tabla 40: Viewer Reports - Caja

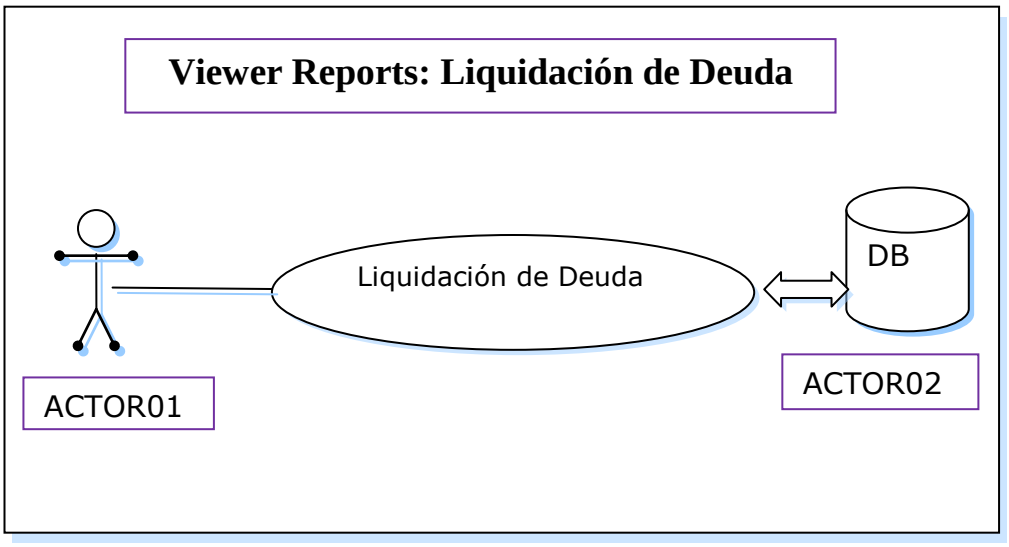


Figura 58: Viewer Reports – Liquidación de Deuda

Caso de Uso	Viewer Reports
Nombre:	Liquidación de Deuda
Actores:	ACTOR 01 – ACTOR02
Precondiciones:	1.- Existan Reportes
Flujo de Trabajo	1.- Hacer doble Click en el reporte 2.- Seleccionar los filtros deseados 3'- Click en Buscar
Comentarios	primordial seleccionar la Fecha

Tabla 41: Descripción - Viewer Reports - Liquidación

Codificación de las estructuras:

Cobranzas.Configuración: Esta capa permite establecer la Conexión y Configuración del Sistema con la base de datos del cliente

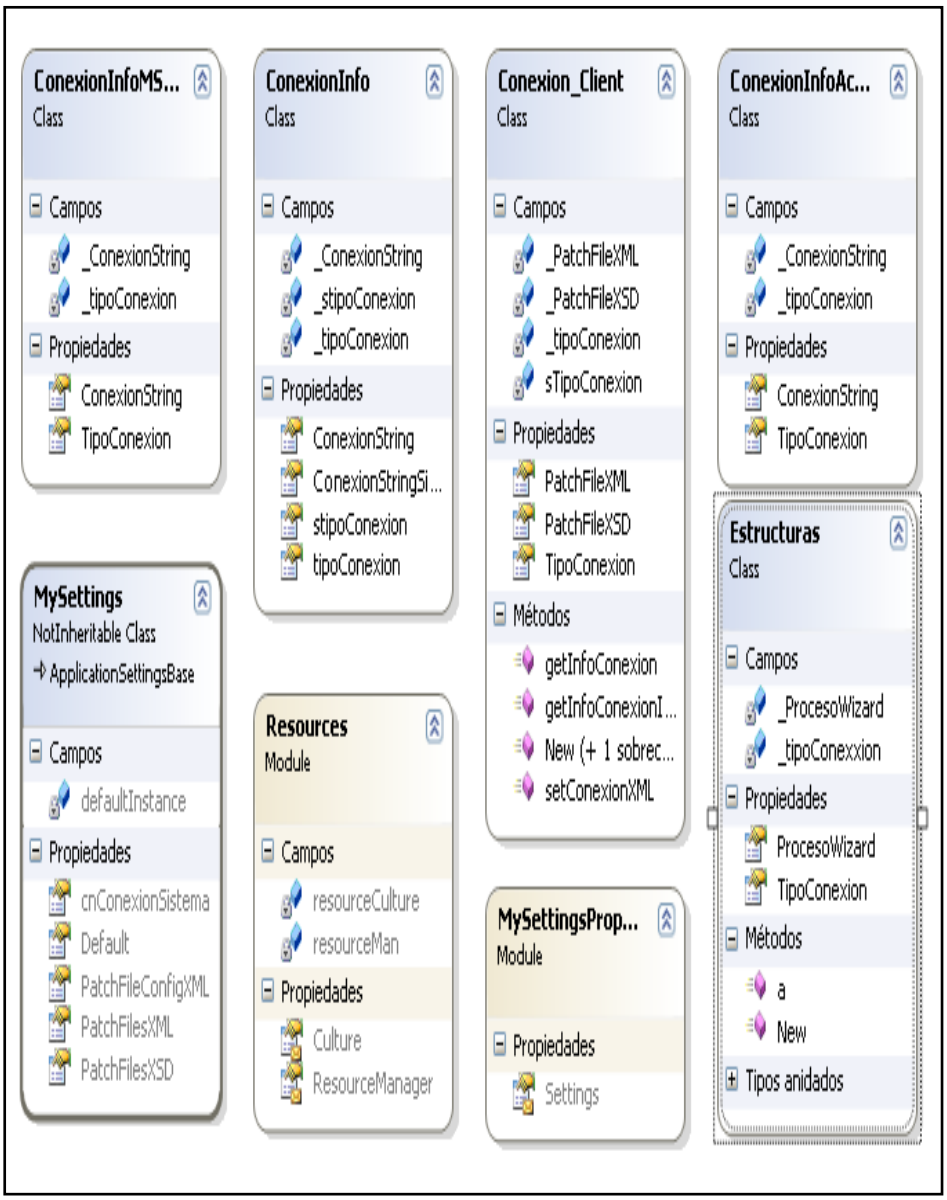


Figura 59: Estructura: Cobranza Configuración

*** ---------- Conexion Client.vb ----------***

Esta clase nos permitira conertarnos con la base de datos del cliente

*** ---------- ConexionInfo ----------***

Esta clase permite establecer la Conexión con nuestra Base de Datos CARTERABD

```
Public Class ConexionInfo
   Private _ConexionString As String
   Public Property ConexionString() As String
      Get
         Return _ConexionString
      End Get
      Set(ByVal value As String)
         _ConexionString = value
      End Set
   End Property
   Private _tipoConexion As Estructuras.Conexiones
   Public Property tipoConexion() As Estructuras.Conexiones
      Get
        Select Case _stipoConexion
            Case "MSSQL" : tipoConexion = Estructuras.Conexiones.MsSQL
            Case "ACCESS" : tipoConexion = Estructuras.Conexiones.Access
            Case "ORACLE" : tipoConexion = Estructuras.Conexiones.Oracle
         End Select
         Return tipoConexion
         End Property
  Private _stipoConexion As String
   Public Property stipoConexion() As String
      Get
         Return _stipoConexion
      End Get
      Set(ByVal value As String)
         _stipoConexion = value
      End Set
   End Property
 Public ReadOnly Property ConexionStringSistema() As String
      Get
         Return "Data Source=localhost;Initial Catalog=CARTERABD;User
Id=sa;password=;" 'My.Settings.cnConexionSistema
      End Get
   End Property
End Class
```

```
* ------------------------------ ConexionInfoAccess ----------------------------------*
Public Class ConexionInfoAccess
    Private _ConexionString As String
    Public Property ConexionString() As String
        Get
            Return _ConexionString
        End Get
        Set(ByVal value As String)
            _ConexionString = value
        End Set
    End Property
    Private _tipoConexion As Estructuras.Conexiones
    Public ReadOnly Property TipoConexion() As Estructuras.Conexiones
        Get
            Return Estructuras.Conexiones.Access
        End Get
    End Property
```

```
* ---------------------------------- ConexionInfoMSSQL ------------------------------*

Imports System.Configuration
Public Class ConexionInfoMSSQL
    Private _ConexionString As String
    Public Property ConexionString() As String
        Get
            Return _ConexionString
        End Get
        Set(ByVal value As String)
            _ConexionString = value
        End Set
    End Property
    Private _tipoConexion As Estructuras.Conexiones
    Public ReadOnly Property TipoConexion() As Estructuras.Conexiones
        Get
            Return Estructuras.Conexiones.MsSQL
        End Get
    End Property
End Class
```

*** ---------- ConexionEstructuras.vb ----------***

```
Public Class Estructuras
    Enum Conexiones
        Access = 1
        MsSQL = 2
        Oracle = 3
    End Enum
    Enum TipoWizard
        Atras = -1
        Siguiente = 1
    End Enum
    Private _tipoConexxion As Conexiones
    Public Property TipoConexion() As Conexiones
        Get
            Return _tipoConexxion
        End Get
        Set(ByVal value As Conexiones)
            _tipoConexxion = value
        End Set
    End Property
    Sub a()
    Dim a As Conexiones
        a = Conexiones.Oracle
    End Sub
    Private _ProcesoWizard As TipoWizard
    Public Property ProcesoWizard() As TipoWizard
        Get
            Return _ProcesoWizard
        End Get
        Set(ByVal value As TipoWizard)
            _ProcesoWizard = value
        End Set
    End Property
    Sub New()
    End Sub
End Class
```

Cobranza.Info: En esta capa se encuentra almacenada toda las estructuras de las clases que van hacer utilizadas en el sistema para el Ambiente Form, Web y Móvil.

Figura60: Estructura: Cobranza Info

*** ---------- CLASES INFO.vb ----------***

C:\Tesis Lourdes\Gestor Cobranza\Cobranza.info\EmpresaInfo.vb

EmpresaInfo.vb

```
Imports System.Xml.Serialization
<Serializable()> _
Public Class EmpresaInfo
    Private _IdEmpresa As String
    Private _fondo As String
    Private _logo As String
    Private _tel_int As String
    Private _imagen As String
    Private _direccion As String
    Private _fax As String
    Private _telefonos As String
    Private _ruc As String
    Private _gerente As String
    Private _nombre As String
    <XmlElement("IdEmpresa")> _
    Public Property IdEmpresa() As String
        Get
            Return _IdEmpresa
        End Get
        Set(ByVal value As String)
            _IdEmpresa = value
        End Set
    End Property
    <XmlElement("nombre")> _
    Public Property nombre() As String
        Get
            Return _nombre
        End Get
        Set(ByVal value As String)
            _nombre = value
        End Set
    End Property
    <XmlElement("gerente")> _
    Public Property gerente() As String
        Get
            Return _gerente
        End Get
        Set(ByVal value As String)
```

```
        _gerente = value
    End Set
End Property
<XmlElement("ruc")> _
Public Property ruc() As String
    Get
        Return _ruc
    End Get
    Set(ByVal value As String)
        _ruc = value
    End Set
End Property
<XmlElement("telefonos")> _
Public Property telefonos() As String
    Get
        Return _telefonos
    End Get
    Set(ByVal value As String)
        _telefonos = value
    End Set
End Property
<XmlElement("fax")> _
Public Property fax() As String
    Get
        Return _fax
    End Get
    Set(ByVal value As String)
        _fax = value
    End Set
End Property
<XmlElement("direccion")> _
Public Property direccion() As String
    Get
        Return _direccion
    End Get
    Set(ByVal value As String)
        _direccion = value
    End Set
End Property
<XmlElement("imagen")> _
Public Property imagen() As String
    Get
        Return _imagen
    End Get
```

```
    Set(ByVal value As String)
        _imagen = value
    End Set
End Property
<XmlElement("tel_int")> _
Public Property tel_int() As String
    Get
        Return _tel_int
    End Get
    Set(ByVal value As String)
        _tel_int = value
    End Set
End Property
<XmlElement("logo")> _
Public Property logo() As String
    Get
        Return _logo
    End Get
    Set(ByVal value As String)
        _logo = value
    End Set
End Property
<XmlElement("fondo")> _
Public Property fondo() As String
    Get
        Return _fondo
    End Get
    Set(ByVal value As String)
        _fondo = value
    End Set
End Property
Sub New()
End Sub
Sub New(ByVal value As String)
    _direccion = value
End Sub
Sub New( _
     ByVal IdEmpresa As Integer _
    , ByVal fondo As String _
    , ByVal logo As String _
    , ByVal tel_int As String _
    , ByVal imagen As String _
    , ByVal direccion As String _
    , ByVal fax As String _
```

```
      , ByVal telefonos As String _
      , ByVal ruc As String _
      , ByVal gerente As String _
      , ByVal nombre As String _
      )
      _IdEmpresa = IdEmpresa
      _fondo = fondo
      _logo = logo
      _tel_int = tel_int
      _imagen = imagen
      _direccion = direccion
      _fax = fax
      _telefonos = telefonos
      _ruc = ruc
      _gerente = gerente
      _nombre = nombre
   End Sub
End Class
```

---------- OTRAS CLASES INFO ----------

- **Grupo_RptInfo.vb**

  La estrucutra de esta clase permitira establecer a que grupo del perfil de usuarios se va a accesar

```
Public Class Grupo_RptInfo
   Private _IdGrupo As Integer
   Private _Descripcion As String
   Public Property IdGrupo() As Integer
      Get
         Return _IdGrupo
      End Get
      Set(ByVal value As Integer)
         _IdGrupo = value
      End Set
   End Property
   Public Property Descripcion() As String
      Get
         Return _Descripcion
```

```
        End Get
        Set(ByVal value As String)
            _Descripcion = value
        End Set
    End Property
    Sub New()
    End Sub
    Sub New( _
                ByVal _IdGrupo As Integer, _
                ByVal _Descripcion As String _
                )
        _IdGrupo = IdGrupo
        _Descripcion = Descripcion
    End Sub
End Class
```

- **HICC_CobrosxClienteInfo.vb**

- Esta clase va permitir guardar el Nombre del reporte y la fecha en que se guardo par luego ser buscada en la consulta historica

- **HistóricoRpt_info**

- En esta clase permitira realizar la busqueda de los reportes que se guardaron

- **ReportesInfo.vb**

- Esta clase permitira escoger el reporte que sera visto dependiendo del grupo de perfil que este tratando de acceder a la programa.

- **MenuInfo.vb**

- Esta clase crea la estructura que tendra el menu para identificar cual de los tres perfiles podra acceder al programa

- **TablaInfo.vb**

- Esta clase permitira selecciona a cada una de las tablas atravez codigo, nombre y el cambpo de la tabla para luego ser relacionadas

- **TablaVsXML.vb**

- Permite relacionar las tablas del cliente Vs las tablas de mis XML

- **UsusariosInfo.vb**

- Clase disebada obtener los perfiles de Ususarios que podran ingresar al programa ingresando el password

- **XMLInfo.vb**

- Esta clase se relacionara con las clases del cliente

Cobranza. Data: esta capa realiza la conexión para enlazarla con la base de datos de todas las funciones, procedimientos y sp. para migrar los datos del cliente, (insertar datos del cliente a nuestra base), esta capa recibe como parámetros los objetos de tipo Info para luego asignarlos y retornarlos n una lista.

Figura 61: Estructutra: Cobranzas.Data

---------- CLASES DATA ----------

D:\Tesis Lourdes\Gestor Cobranza\Cobranza.Data\EmpresaData.vb

spRpt_ProcesaData_CobrosxCliente_Data.vb

```
Imports Cobranza.info
Imports System.Data.SqlClient
Impots Cobranza.Configuracion
Imports System.Data.OleDb
Public Class spRpt_ProcesaData_CobrosXClienteData
   Public Sub EjecutarRpt1(ByVal idEmpreas As Integer, ByVal idSucu As Integer,
ByVal idCobroIni As String, ByVal idCobroFin As String _
                        , ByVal idClie As Integer, ByVal idVen As Integer, ByVal fechaIni
As Date, ByVal fechaFin As Date _
                        , ByVal idTipoEstadoIni As String, ByVal idTipoEstadoFin As
String _
                         , ByVal TipoRpt As String, ByVal NSucursal As String, ByVal
NClie As String, ByVal NVend As String, _
                         ByVal NTipoRpt As String, ByVal NTipoCobro As String _
                         , ByVal NameReport As String, ByVal idRpthistory As String)
      Try
         Dim Conn As SqlConnection
         Conn = DataConexion.GetConexion
         Dim cm As New
SqlClient.SqlCommand("spRpt_ProcesaData_CobrosXCliente", Conn)
         Dim idClienteIni As Integer
         Dim idClienteFin As Integer
         Dim idVendedorIni As Integer
         Dim idVendedorFin As Integer
         Dim lidsucuIni As Integer
         Dim lidsucuFin As Integer
         If idClie = 0 Then
            idClienteIni = 1
            idClienteFin = 999999
         Else
            idClienteIni = idClie
            idClienteFin = idClie
         End If
         If idVen = 0 Then
            idVendedorIni = 1
            idVendedorFin = 999
         Else
            idVendedorIni = idVen
```

```
            idVendedorFin = idVen
        End If
        If idCobroIni = "All" Then
            idCobroIni = "AAA"
            idCobroFin = "ZZZ"
        End If
        If idCobroIni = Nothing Or idCobroFin = Nothing Then
            idCobroIni = "AAA"
            idCobroFin = "ZZZ"
        End If
        If idSucu = 0 Then
            lidsucuIni = 1
            lidsucuFin = 2
        Else
            lidsucuIni = idSucu
            lidsucuFin = idSucu
        End If
        If idTipoEstadoIni = "" Then
            idTipoEstadoIni = "A"
            idTipoEstadoFin = "Z"
        End If
        cm.Parameters.Add("@i_cia", SqlDbType.Int).Value = idEmpreas
        cm.Parameters.Add("@i_sucuIni", SqlDbType.Int).Value = lidsucuIni
        cm.Parameters.Add("@i_sucuFin", SqlDbType.Int).Value = lidsucuFin
        cm.Parameters.Add("@i_idFormaIni", SqlDbType.NChar, 3).Value =
idCobroIni
        cm.Parameters.Add("@i_idFormaFin", SqlDbType.NChar, 3).Value =
idCobroFin
        cm.Parameters.Add("@i_clienteIni", SqlDbType.Int).Value = idClienteIni
        cm.Parameters.Add("@i_clienteFin", SqlDbType.Int).Value = idClienteFin
        cm.Parameters.Add("@i_vendIni", SqlDbType.Int).Value = idVendedorIni
        cm.Parameters.Add("@i_vendFin", SqlDbType.Int).Value = idVendedorFin
        cm.Parameters.Add("@i_fechaIni", SqlDbType.Date).Value = fechaIni
        cm.Parameters.Add("@i_fechaFin", SqlDbType.Date).Value = fechaFin
        cm.Parameters.Add("@i_EstadoIni", SqlDbType.NChar, 3).Value =
idTipoEstadoIni
        cm.Parameters.Add("@i_EstadoFin", SqlDbType.NChar, 3).Value =
idTipoEstadoFin
        cm.Parameters.Add("@i_TipoRpt", SqlDbType.Char, 20).Value = TipoRpt
        cm.Parameters.Add("@PSucursal", SqlDbType.Char, 20).Value = NSucursal
        cm.Parameters.Add("@PCliente", SqlDbType.Char, 20).Value = NClie
        cm.Parameters.Add("@PVendedor", SqlDbType.Char, 20).Value = NVend
        cm.Parameters.Add("@PTipoRpt", SqlDbType.Char, 20).Value = NTipoRpt
```

```
        cm.Parameters.Add("@PTipoCobro", SqlDbType.Char, 20).Value =
NTipoCobro
        cm.Parameters.Add("@i_NameReports", SqlDbType.NChar, 50).Value =
NameReport
        cm.Parameters.Add("@i_idRpthistory", SqlDbType.NChar, 100).Value =
idRpthistory
        cm.CommandType = CommandType.StoredProcedure
        cm.Connection.Open()
        cm.ExecuteScalar()
        cm.Connection.Close()
        cm.Dispose()
      Catch ex As Exception
      End Try
   End Sub
   Public Sub EjecutarCancelacionxFacturas(ByVal idEmpreas As Integer, ByVal
idSucu As Integer, ByVal idCobroIni As String, ByVal idCobroFin As String _
                        , ByVal idClie As Integer, ByVal idVen As Integer, ByVal fechaIni
As Date, ByVal fechaFin As Date _
                        , ByVal idTipoEstadoIni As String, ByVal idTipoEstadoFin As
String _
                        , ByVal NumFacturaIni As Double, ByVal NumFacturaFin As
Double _
                         , ByVal TipoRpt As String, ByVal NSucursal As String, ByVal
NClie As String, ByVal NVend As String, _
                         ByVal NTipoRpt As String, ByVal NTipoCobro As String _
                         , ByVal NameReport As String, ByVal idRpthistory As String)
      Try
        Dim Conn As SqlConnection
        Conn = DataConexion.GetConexion
        Dim cm As New
SqlClient.SqlCommand("spRpt_ProcesaData_CancelacionxFacturas", Conn)
        Dim idClienteIni As Integer
        Dim idClienteFin As Integer
        Dim idVendedorIni As Integer
        Dim idVendedorFin As Integer
          Dim lidsucuIni As Integer
        Dim lidsucuFin As Integer
        If idClie = 0 Then
          idClienteIni = 1
          idClienteFin = 999999
        Else
          idClienteIni = idClie
          idClienteFin = idClie
        End If
```

```
        If idVen = 0 Then
            idVendedorIni = 1
            idVendedorFin = 999
        Else
            idVendedorIni = idVen
            idVendedorFin = idVen
        End If
        If idCobroIni = "All" Then
            idCobroIni = "AAA"
            idCobroFin = "ZZZ"
        End If
        If idCobroIni Is Nothing Or idCobroFin Is Nothing Then
            idCobroIni = "AAA"
            idCobroFin = "ZZZ"
        End If
        If idSucu = 0 Then
            lidsucuIni = 1
            lidsucuFin = 2
        Else
            lidsucuIni = idSucu
            lidsucuFin = idSucu
        End If
        If idTipoEstadoIni = "" Then
            idTipoEstadoIni = "A"
            idTipoEstadoFin = "Z"
        End If
        If NumFacturaIni = 0 Then
            NumFacturaIni = 0
            NumFacturaFin = 500000
        End If
        cm.Parameters.Add("@i_cia", SqlDbType.Int).Value = idEmpreas
        cm.Parameters.Add("@i_sucuIni", SqlDbType.Int).Value = lidsucuIni
        cm.Parameters.Add("@i_sucuFin", SqlDbType.Int).Value = lidsucuFin
        cm.Parameters.Add("@i_idFormaIni", SqlDbType.NChar, 3).Value =
idCobroIni
        cm.Parameters.Add("@i_idFormaFin", SqlDbType.NChar, 3).Value =
idCobroFin
        cm.Parameters.Add("@i_clienteIni", SqlDbType.Int).Value = idClienteIni
        cm.Parameters.Add("@i_clienteFin", SqlDbType.Int).Value = idClienteFin
        cm.Parameters.Add("@i_vendIni", SqlDbType.Int).Value = idVendedorIni
        cm.Parameters.Add("@i_vendFin", SqlDbType.Int).Value = idVendedorFin
        cm.Parameters.Add("@i_fechaIni", SqlDbType.Date).Value = fechaIni
        cm.Parameters.Add("@i_fechaFin", SqlDbType.Date).Value = fechaFin
```

```
        cm.Parameters.Add("@i_EstadoIni", SqlDbType.NChar, 3).Value =
idTipoEstadoIni
        cm.Parameters.Add("@i_EstadoFin", SqlDbType.NChar, 3).Value =
idTipoEstadoFin
        cm.Parameters.Add("@i_NumFacturaIni", SqlDbType.Float).Value =
NumFacturaIni
        cm.Parameters.Add("@i_NumFacturaFin", SqlDbType.Float).Value =
NumFacturaFin
        cm.Parameters.Add("@i_TipoRpt", SqlDbType.Char, 20).Value = TipoRpt
        cm.Parameters.Add("@i_idRpthistory", SqlDbType.NChar, 100).Value =
idRpthistory
        cm.Parameters.Add("@i_NameReports", SqlDbType.NChar, 50).Value =
NameReport
        cm.Parameters.Add("@PSucursal", SqlDbType.Char, 20).Value = NSucursal
        cm.Parameters.Add("@PCliente", SqlDbType.Char, 20).Value = NClie
        cm.Parameters.Add("@PVendedor", SqlDbType.Char, 20).Value = NVend
        cm.Parameters.Add("@PTipoRpt", SqlDbType.Char, 20).Value = NTipoRpt
        cm.Parameters.Add("@PTipoCobro", SqlDbType.Char, 20).Value =
NTipoCobro
        cm.CommandType = CommandType.StoredProcedure
        cm.Connection.Open()
        cm.ExecuteScalar()
        cm.Connection.Close()
        cm.Dispose()
    Catch ex As Exception
    End Try
  End Sub
  Public Sub EjecutarRptXProducto(ByVal idEmpreas As Integer, ByVal idSucuIni
As Integer, ByVal idSucuFin As Integer, ByVal idCobroIni As String, ByVal
idCobroFin As String _
                    , ByVal idClie As Integer, ByVal idVen As Integer, ByVal fechaIni
As Date, ByVal fechaFin As Date _
                    , ByVal ProductoIni As Double, ByVal ProductoFin As Double _
                    , ByVal TipoRpt As String, ByVal NSucursal As String, ByVal
NClie As String, ByVal NVend As String _
                    , ByVal NTipoRpt As String, ByVal NTipoCobro As String, ByVal
NameReport As String _
                    , ByVal idRpthistory As Double)
    Try
        Dim Conn As SqlConnection
        Conn = DataConexion.GetConexion
        Dim cm As New
SqlClient.SqlCommand("spRpt_ProcesaData_CobrosXProducto", Conn)
        Dim idClienteIni As Integer
```

```
        Dim idClienteFin As Integer
        Dim idVendedorIni As Integer
        Dim idVendedorFin As Integer
            If idClie = 0 Then
           idClienteIni = 1
           idClienteFin = 999999
        Else
           idClienteIni = idClie
           idClienteFin = idClie
        End If
        If idVen = 0 Then
           idVendedorIni = 1
           idVendedorFin = 999
        Else
           idVendedorIni = idVen
           idVendedorFin = idVen
        End If
        If idCobroIni = "All" Then
           idCobroIni = "AAA"
           idCobroFin = "ZZZ"
        End If
        If idCobroIni Is Nothing Or idCobroFin Is Nothing Then
           idCobroIni = "AAA"
           idCobroFin = "ZZZ"
        End If
        cm.Parameters.Add("@i_cia", SqlDbType.Int).Value = idEmpreas
        cm.Parameters.Add("@i_sucuIni", SqlDbType.Int).Value = idSucuIni
        cm.Parameters.Add("@i_sucuFin", SqlDbType.Int).Value = idSucuFin
        cm.Parameters.Add("@i_idFormaIni", SqlDbType.NChar, 3).Value =
idCobroIni
        cm.Parameters.Add("@i_idFormaFin", SqlDbType.NChar, 3).Value =
idCobroFin
        cm.Parameters.Add("@i_clienteIni", SqlDbType.Int).Value = idClienteIni
        cm.Parameters.Add("@i_clienteFin", SqlDbType.Int).Value = idClienteFin
        cm.Parameters.Add("@i_vendIni", SqlDbType.Int).Value = idVendedorIni
        cm.Parameters.Add("@i_vendFin", SqlDbType.Int).Value = idVendedorFin
        cm.Parameters.Add("@i_fechaIni", SqlDbType.Date).Value = fechaIni
        cm.Parameters.Add("@i_fechaFin", SqlDbType.Date).Value = fechaFin
        cm.Parameters.Add("@i_TipoRpt", SqlDbType.Char, 20).Value = TipoRpt
        cm.Parameters.Add("@PSucursal", SqlDbType.Char, 20).Value = NSucursal
        cm.Parameters.Add("@PCliente", SqlDbType.Char, 20).Value = NClie
        cm.Parameters.Add("@PVendedor", SqlDbType.Char, 20).Value = NVend
        cm.Parameters.Add("@PTipoRpt", SqlDbType.Char, 20).Value = NTipoRpt
```

```
            cm.Parameters.Add("@PTipoCobro", SqlDbType.Char, 20).Value =
NTipoCobro
            cm.Parameters.Add("@i_NameReports", SqlDbType.NChar, 50).Value =
NameReport
            cm.Parameters.Add("@i_ProductoIni", SqlDbType.Float, 50).Value =
ProductoIni
            cm.Parameters.Add("@i_ProductoFin", SqlDbType.Float, 50).Value =
ProductoFin
            cm.Parameters.Add("@i_idRpthistory", SqlDbType.Float).Value =
idRpthistory
            cm.CommandType = CommandType.StoredProcedure
            cm.Connection.Open()
            cm.ExecuteScalar()
            cm.Connection.Close()
            cm.Dispose()
        Catch ex As Exception
        End Try
    End Sub
    Public Function getListaRptHist() As List(Of HistoricoRpt_info)
        Dim dr As SqlDataReader
        Dim Conn As SqlConnection
        Dim comand As SqlClient.SqlCommand
        Dim instruccion As String
        Dim oHRpt As New HistoricoRpt_info
        Dim listHrpt As New List(Of HistoricoRpt_info)
        Try
            Conn = DataConexion.GetConexion
            instruccion = "select distinct cr_idProceso ,cr_NameReports
,cr_fechaProceGuardarRpt  from hiCC_CobrosxCliente "
            comand = New SqlClient.SqlCommand(instruccion, Conn)
            Conn.Open()
            dr = comand.ExecuteReader
            While dr.Read
                oHRpt = New HistoricoRpt_info
                oHRpt.IdProceso = dr("cr_idProceso")
                oHRpt.NombreRpt = dr("cr_NameReports")
                oHRpt.FechaProceso = dr("cr_fechaProceGuardarRpt")
                listHrpt.Add(oHRpt)
            End While
            Conn.Close()
            Return listHrpt
        Catch ex As Exception
            Return listHrpt
            comand.Dispose()
```

```
        End Try
    End Function
    Public Function getListaRptHistXProducto() As List(Of HistoricoRpt_info)
        Dim dr As SqlDataReader
        Dim Conn As SqlConnection
        Dim comand As SqlClient.SqlCommand
        Dim instruccion As String
        Dim oHRpt As New HistoricoRpt_info
        Dim listHrpt As New List(Of HistoricoRpt_info)
        Try
            Conn = DataConexion.GetConexion
            instruccion = "select distinct cr_idProceso ,cr_NameReports
,cr_fechaProceGuardarRpt  from HICC_CobrosxProducto "
            comand = New SqlClient.SqlCommand(instruccion, Conn)
            Conn.Open()
            dr = comand.ExecuteReader
            While dr.Read
                oHRpt = New HistoricoRpt_info
                oHRpt.IdProceso = dr("cr_idProceso")
                oHRpt.NombreRpt = dr("cr_NameReports")
                oHRpt.FechaProceso = dr("cr_fechaProceGuardarRpt")
                listHrpt.Add(oHRpt)
            End While
            Conn.Close()
            Return listHrpt
        Catch ex As Exception
            Return listHrpt
            comand.Dispose()
        End Try
    End Function
End Class
```

CobranzaBussiness: Esta capa se va a encargar de subir todos los métodos que están en las clases Data y la Clase Info: CargarData_MGProduccion, Insert_TableMG (migrar las tablas), serializarlas, buscar_ColRelacionada, Convertir_TablaInfo, SetearCamposTabla_Info, ConvertirData, Client_Info, ComparaciónSchema, Serializar, De serializar

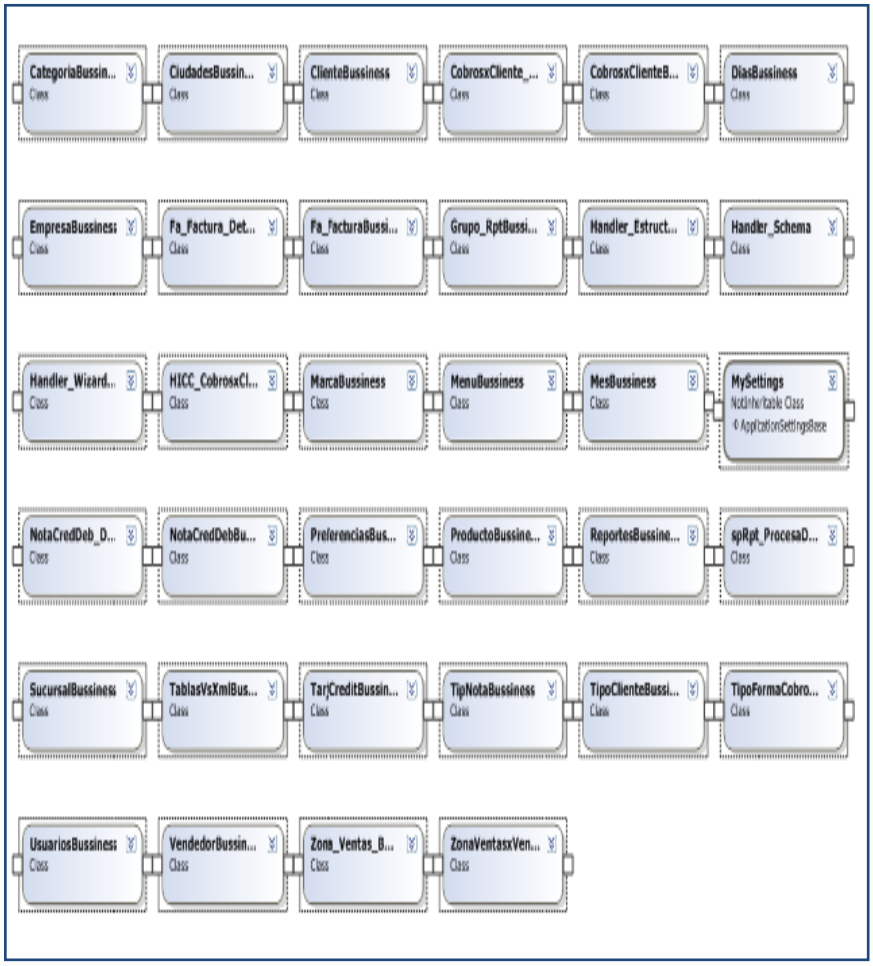


Figura 62: Estructura: CobranzaBussiness

*** ----------CODIFICACION CLASES BUSSINESS.vb ----------***

D:\Tesis Lourdes\Gestor Cobranza\Cobranza.bussiness\EmpresaBussiness.vb

ClienteBussiness.vb

```
Imports Cobranza.info
Imports Cobranza.Data
Imports System.Reflection
Imports System.Runtime.Serialization.Formatters.Binary
Imports System.Xml.Serialization
Imports System.IO
Imports System.Xml
Public Class ClienteBussiness
    Dim tablasvsXml As List(Of TablasVsXML)
    Dim tablasvsXMLBu As New TablasVsXmlBussines
    Dim HSchema As New Handler_Schema
    Sub New()
        tablasvsXml = tablasvsXMLBu.CargarListaRelaciones("Cliente")
    End Sub
    Sub New(ByVal IniciarCt As Integer)

    End Sub
    Public Function CargaDataMG_Produccion()
        Dim odata As New Data_Comands_Exec
        Return odata.Execute_Comands("spMG_Carga_Validacion_Cliente")
    End Function
    '*******MIGRACION A TABLAS MG*****
    Public Sub Insert_TableMG(ByVal lCliente As List(Of ClienteInfoS))
        Dim clienteD As New ClienteData
        clienteD.Insert_DataInfo_TableMG(lCliente)
    End Sub
    '*****PROCESO DE SETEO PARA SEREALIZACION
    Public Function getListaClienteDS(ByVal Querry As String) As DataSet
        Dim clienteD As New ClienteData
        Dim Ds As New DataSet
        Ds = clienteD.getClienteDS(Querry)
        Return Ds
    End Function
    Public Function NombreCliente(ByVal Cia As String, ByVal Sucu As String) As
List(Of ClienteInfo)
        Dim oCliente As New ClienteData
        Return oCliente.getClienteDS_DBCartera(Cia, Sucu)
    End Function
```

```
    Public Function buscar_ColRelacionada(ByVal ColumnaTabla As String) As
String
        Dim nameColXML As String = ""
        For Each Item As TablasVsXML In tablasvsXml
            If Item.idCampoTabla.Trim = ColumnaTabla.Trim Then
                nameColXML = Item.IdCampoXml.Trim
                Exit For
            End If
        Next
        Return nameColXML
    End Function
    Public Function ConvertirTabla_Info(ByVal DSCliente As DataSet) As List(Of
ClienteInfoS)
        Dim ClienteD As New ClienteData
        Dim xmlT As New TablasVsXmlBussines
        Dim listaCliente As New List(Of ClienteInfoS)
        Dim oCliente As ClienteInfoS
        Dim indexCol As Integer = 0
        Dim colInfo As String
        Dim colTable As String
        Dim valor As String
        'Dim i As Integer = 0
        For Each iRow As DataRow In DSCliente.Tables(0).Rows
            oCliente = New ClienteInfoS("", "", "", "", "", "", "", "", "", "", "", "", "", "", "",
"", "", "", "", "", "", "", "", "")
            For Each iCol As DataColumn In DSCliente.Tables(0).Columns
                colTable = iCol.ColumnName.Trim
                colInfo = buscar_ColRelacionada(colTable)
                valor = iRow(colTable).ToString
                SetearCamposTabla_Info(colInfo, valor, oCliente)
            Next
            listaCliente.Add(oCliente)
            'i += 1
            'If i = 100 Then
            '    Exit For
            'End If
        Next
        Return listaCliente
    End Function
    Sub SetearCamposTabla_Info(ByVal idCampoInfo As String, ByVal valor As
String, ByVal oCliente As ClienteInfoS)
        ' declaracion del tipo
        Dim t As Type = GetType(ClienteInfoS)
        'llamada al constructor para instanciar
```

```
Dim c As ConstructorInfo = t.GetConstructor(New Type() {GetType(String)})
"invocando al constructor y enviando el parametro de entrada del constructor
Dim sb As Object = c.Invoke(New Object() {"iniciar"})
Dim p() As PropertyInfo = t.GetProperties '(idCampoInfo)
For i As Integer = 0 To t.GetProperties.Length - 1
   Dim CampoInfo As String = p(i).Name.Trim
   If idCampoInfo = CampoInfo Then
      Select Case CampoInfo
         Case "IdCompania"
            oCliente.IdCompania = valor
         Case "IdSucursal"
            oCliente.IdSucursal = valor
         Case "IdCliente"
            oCliente.IdCliente = valor
         Case "tipoDoc"
            oCliente.tipoDoc = valor
         Case "cedula_ruc_Pass"
            oCliente.cedula_ruc_Pass = valor
         Case "nombre"
            oCliente.nombre = valor
         Case "RazonSocial"
            oCliente.RazonSocial = valor
         Case "direccion"
            oCliente.direccion = valor
         Case "telefono"
            oCliente.telefono = valor
         Case "email"
            oCliente.email = valor
         Case "tipo_cliente"
            oCliente.tipo_cliente = valor
         Case "credito"
            oCliente.credito = valor
         Case "plazo"
            oCliente.plazo = valor
         Case "observacion"
            oCliente.observacion = valor
         Case "fecha"
            oCliente.fecha = valor
         Case "IdCiudad"
            oCliente.IdCiudad = valor
         Case "Idzona"
            oCliente.Idzona = valor
         Case "cupo"
            oCliente.cupo = valor
```

```
                Case "casilla"
                    oCliente.casilla = valor
                Case "fax"
                    oCliente.fax = valor
                Case "IdActividad"
                    oCliente.IdActividad = valor
                Case "estado"
                    oCliente.estado = valor
            End Select
        End If
    Next
End Sub
'****FIN DE SEREALIZACION
'****SEREALIZACION Y COMPARACION CON SCHEMA
Public Function ConvertDataClient_Info(ByVal Querry As String) As List(Of
ClienteInfoS)
    'OBTENCION DE INFO PARA SEREALIZAR
    Dim lCliente As New List(Of ClienteInfoS)
    Dim dsE As New DataSet
    dsE = getListaClienteDS(Querry) 'para el dataset
    lCliente = ConvertirTabla_Info(dsE) ' para la lista
    Return lCliente
End Function
Public Function ComparacionSchema(ByVal PathXml As String, ByVal PathXsd
As String, ByRef msgExp As String) As Boolean
    If HSchema.EsValidoSchema(PathXsd, PathXml, msgExp) Then
        Return True
    Else
        Return False
    End If
End Function
Public Sub serializar(ByVal patchXML As String, ByVal lCliente As List(Of
ClienteInfoS))
    'SEREALIZACION
    Dim fs As New FileStream(patchXML, FileMode.Create)
    Dim xs As New XmlSerializer(GetType(List(Of ClienteInfoS)))
    xs.Serialize(fs, lCliente)
    fs.Close()
End Sub
Private Sub deserealizarXML(ByVal patchXML As String)
    Dim fs As New FileStream(patchXML, FileMode.Open)
    Dim lCliente As List(Of ClienteInfo)
    Try
        Dim xsCliente As New XmlSerializer(GetType(List(Of ClienteInfo)))
```

```
            lCliente = CType(xsCliente.Deserialize(fs), List(Of ClienteInfo))
            fs.Close()
        Catch ex As Exception
        End Try
    End Sub
    '****FIN SEREALIZACION Y COMPARACION
End Class
```

---------- OTRAS CLASES BUSSINESS ----------

HICC_CobrosxClienteBussiness

```
Imports Cobranza.Data
Imports Cobranza.info
Public Class HICC_CobrosxClienteBussiness
    Public Function DatosHistoricos() As List(Of HICC_CobrosxClienteInfo)
        Dim oHICC_CxCData As New HICC_CobrosxClienteData
        Return oHICC_CxCData.DataHistorico
    End Function
End Class
```

ReportesBussiness

```
Imports Cobranza.info
Imports Cobranza.Data
Public Class ReportesBussiness
    Public Function ListaReportesxViewer(ByVal Querry As String) As List(Of
ReportesInfo)
        Dim oReporteData As New ReportesData
        Return oReporteData.getListReport(Querry)
    End Function
    Public Function ListaDatosReportes() As List(Of ReportesInfo)
        Dim oReportData As New ReportesData
        Return oReportData.getListDataRpt()
    End Function
    Public Function InsertaReporte(ByVal oReporte As ReportesInfo) As Boolean
        Dim ReporteData As New ReportesData
        Return ReporteData.insertReporte(oReporte)
    End Function
    Public Function ModificaReporte(ByVal oReporte As ReportesInfo) As Boolean
        Dim ReporteData As New ReportesData
        Return ReporteData.UpdateReport(oReporte)
    End Function
```

```
    Public Function ValidaReporte(ByVal oReport As ReportesInfo) As List(Of
ReportesInfo)
        Dim ReporteData As New ReportesData
        Return ReporteData.getDataReporte(oReport)
    End Function
    Public Function OptenerDatosReporte(ByVal NameReport As String) As
ReportesInfo
        Dim ReporteData As New ReportesData
        Return ReporteData.getInfoReporte(NameReport)
    End Function
    Public Function CargaReportexModificacion(ByVal oReporte As ReportesInfo) As
List(Of ReportesInfo)
        Dim ReporteData As New ReportesData
        Return ReporteData.getDataReporte(oReporte)
    End Function
    Public Function EliminaReporte(ByVal oReporte As ReportesInfo) As Boolean
        Dim ReporteData As New ReportesData
        Return ReporteData.DeleteReporte(oReporte)
    End Function
End Class
```

spRpt_Procesa Data_CobroscCliente.Bussiness

```
Imports Cobranza.info
Imports Cobranza.Data
Public Class spRpt_ProcesaData_CobrosXClienteBussines
    Public Sub EjecutarRpt1(ByVal idEmpreas As Integer, ByVal idSucu As Integer,
ByVal idCobroIni As String, ByVal idCobroFin As String _
                    , ByVal idClie As Integer, ByVal idVen As Integer, ByVal fechaIni
As Date, ByVal fechaFin As Date _
                    , ByVal idEstadoIni As String, ByVal idEstadoFin As String _
                     , ByVal TipoRpt As String, ByVal NSucursal As String, ByVal
NClie As String, ByVal NVend As String, ByVal NTipoRpt As String, ByVal
NTipoCobro As String _
                     , ByVal NameReport As String, ByVal idRptHist As Double)
        Dim execRptData As New spRpt_ProcesaData_CobrosXClienteData
        execRptData.EjecutarRpt1(idEmpreas, idSucu, idCobroIni, idCobroFin, idClie,
idVen, fechaIni, fechaFin, idEstadoIni, idEstadoFin, TipoRpt, NSucursal, NClie,
NVend, NTipoRpt, NTipoCobro, NameReport, idRptHist)
    End Sub
 Public Sub EjecutarDataRptLiquidacionDeuda(ByVal idEmpreas As Integer, ByVal
idSucu As Integer, ByVal idCobroIni As String, ByVal idCobroFin As String _
```

```
                , ByVal idClie As Integer, ByVal idVen As Integer, ByVal fechaIni
As Date, ByVal fechaFin As Date _
                , ByVal idEstadoIni As String, ByVal idEstadoFin As String _
                , ByVal NumFacIni As Double, ByVal NumFacFin As Double _
                , ByVal TipoRpt As String, ByVal NSucursal As String, ByVal
NClie As String, ByVal NVend As String, ByVal NTipoRpt As String, ByVal
NTipoCobro As String _
                , ByVal NameReport As String, ByVal idRptHist As Double)
        Dim execRptData As New spRpt_ProcesaData_CobrosXClienteData
        execRptData.EjecutarCancelacionxFacturas(idEmpreas, idSucu, idCobroIni,
idCobroFin, _
                                                 idClie, idVen, fechaIni, fechaFin, idEstadoIni,
idEstadoFin, NumFacIni, NumFacFin, TipoRpt, _
                                                 NSucursal, NClie, NVend, NTipoRpt, NTipoCobro,
NameReport, idRptHist)
    End Sub
    Public Sub EjecutarRptxProducto(ByVal idEmpreas As Integer, ByVal idSucuIni
As Integer, ByVal idSucuFin As Integer, ByVal idCobroIni As String, ByVal
idCobroFin As String _
                , ByVal idClie As Integer, ByVal idVen As Integer, ByVal fechaIni
As Date, ByVal fechaFin As Date _
                , ByVal ProductoIni As Double, ByVal Productofin As Double _
                , ByVal TipoRpt As String, ByVal NSucursal As String, ByVal
NClie As String, ByVal NVend As String, ByVal NTipoRpt As String, ByVal
NTipoCobro As String _
                , ByVal NameReport As String, ByVal idRpthistory As String)
        Dim execRptData As New spRpt_ProcesaData_CobrosXClienteData
        execRptData.EjecutarRptXProducto(idEmpreas, idSucuIni, idSucuFin,
idCobroIni, idCobroFin, idClie, idVen, fechaIni, fechaFin, ProductoIni, Productofin,
TipoRpt, NSucursal, NClie, NVend, NTipoRpt, NTipoCobro, NameReport,
idRpthistory)
    End Sub
    Public Function ObtenerListaRptHist() As List(Of HistoricoRpt_info)
        Dim execRptData As New spRpt_ProcesaData_CobrosXClienteData
        Return execRptData.getListaRptHist
    End Function
    Public Function ObtenerListaRptHistxProducto() As List(Of HistoricoRpt_info)
        Dim execRptData As New spRpt_ProcesaData_CobrosXClienteData
        Return execRptData.getListaRptHistXProducto
    End Function
End Class
```

CobranzasSchema.vb

Los Esquemas XML serviran para extraer los datos de los campos de la data que fueron seleccionados en el vinculador de las tablas realcionadas.

---------- CLASES SCHEMA XML ----------

Cobranza.XML

D:\TesisLourdes\GestorCobranza\Cobranza.Schema\empresa.xml

empresa.xml :

```
<?xml version="1.0"?>
<ArrayOfEmpresaInfoS xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <EmpresaInfoS>
    <IdEmpresa>1</IdEmpresa>
    <nombre>EXPOMEDIOS</nombre>
    <gerente>Hernandez Valdiviezo Douglas</gerente>
    <ruc />
    <telefonos />
    <fax />
    <direccion />
    <imagen />
    <tel_int />
    <logo />
    <fondo />
  </ArrayOfEmpresaInfoS>
```

Sucursal.xml:

```
<?xml version="1.0"?>
<ArrayOfSucursalInfoS xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <SucursalInfoS>
    <Idcompania>2</Idcompania>
    <IdSucursal>1</IdSucursal>
    <Codigo>001</Codigo>
    <nombre>GUAYAQUIL - Matriz</nombre>
    <Nemónico />
    <ruc />
```

*** ---------- CLASES SCHEMA XSD ----------***

Cobranza.Schema XSD: Los Esquemas XSD serviran como definicion para validar que los datos que estamos extrayendo esten correctos según el estandar de la base

Cobranzas.XSD

D:\Tesis Lourdes\Gestor Cobranza\Cobranza.Schema\empresa.xml

empresa.xsd

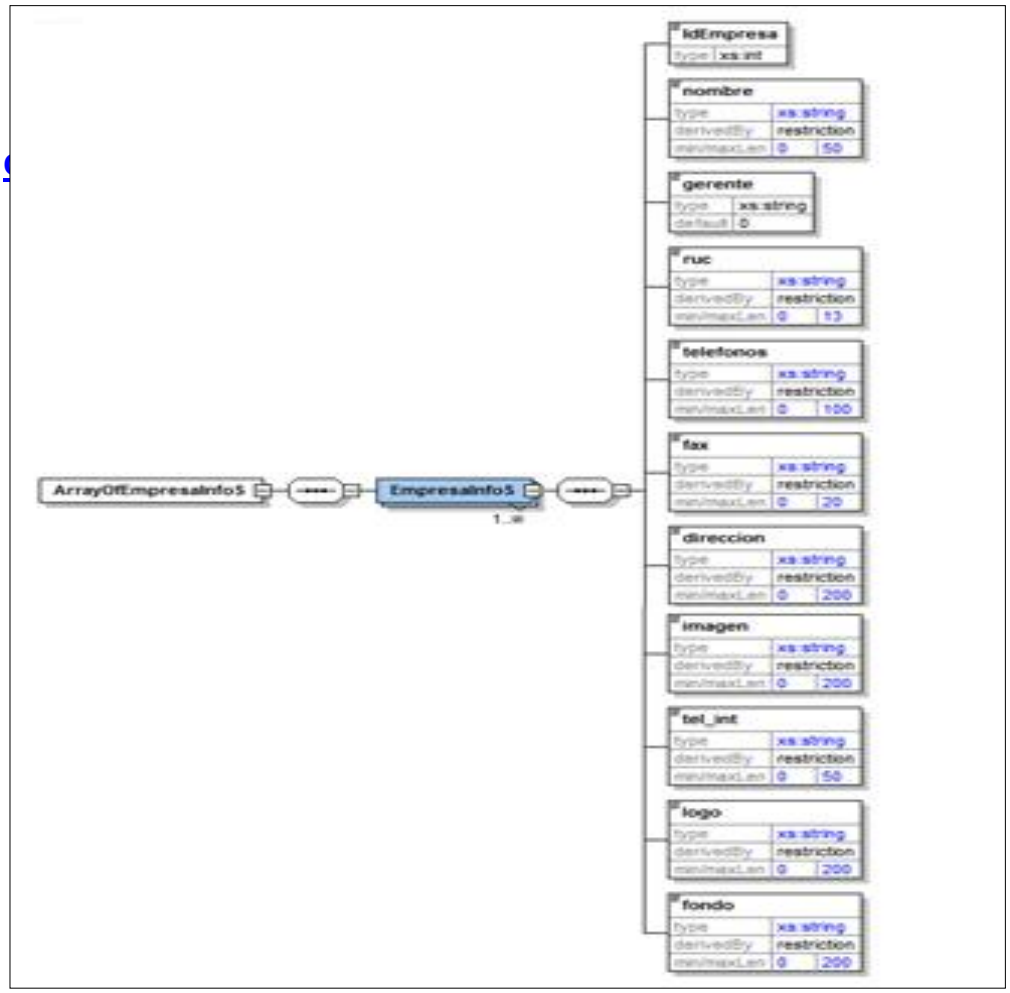


Fig.2.2.1.5.1 Diagrama Schema Empresae.XSD

5.2.2 Cliente.xsd

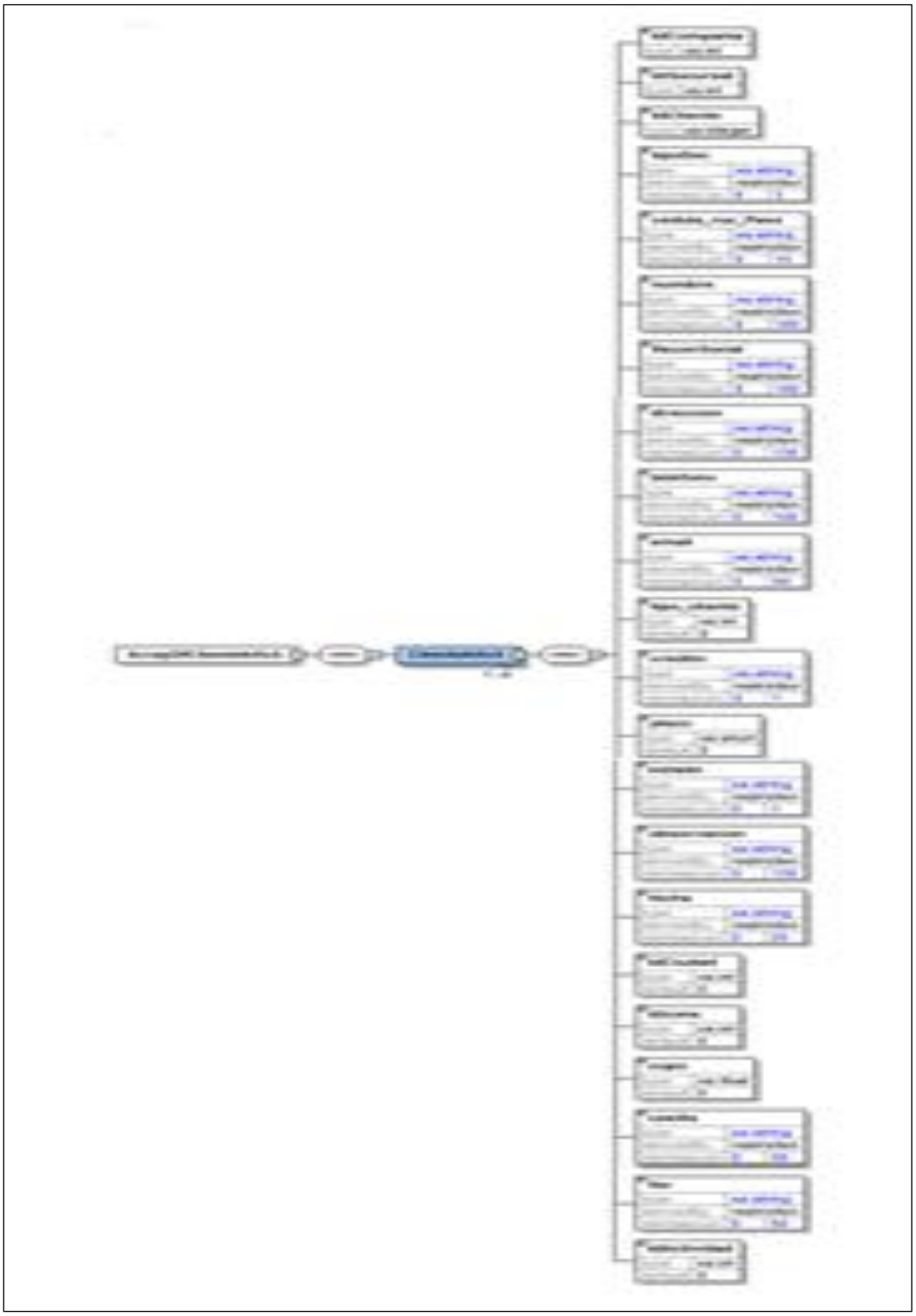
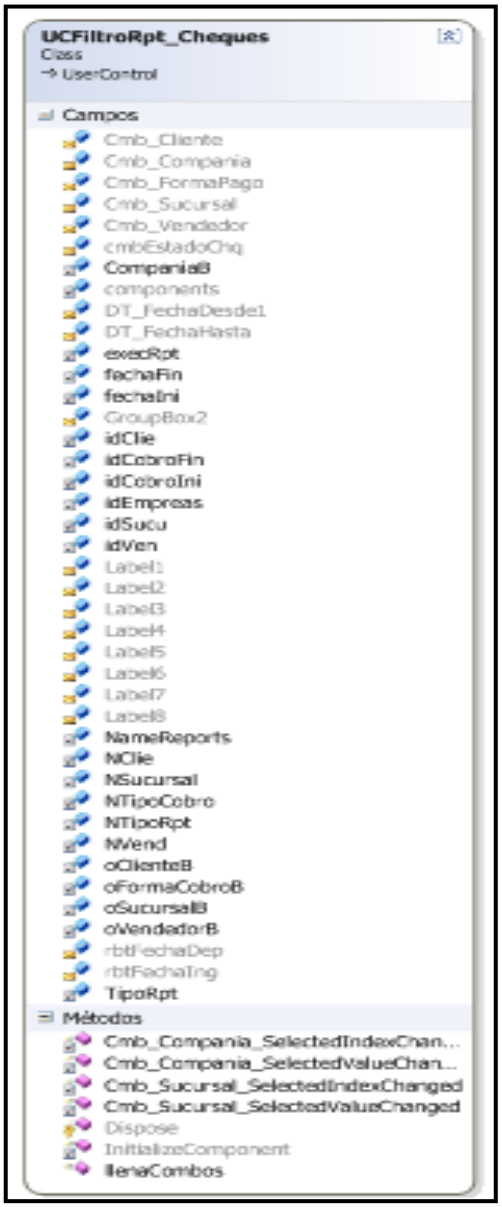


Fig.2.2.1.5.2: Diagrama Schema Cliente.XSD

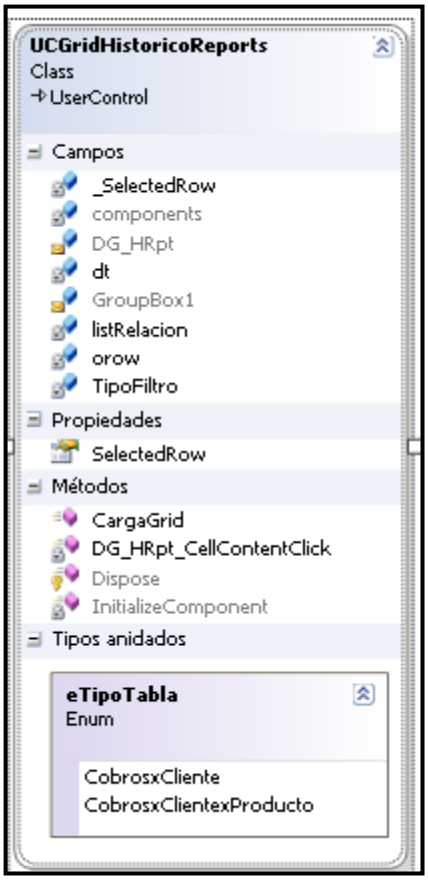
Configuración Modulo**Winform:**

Esta capa sera utilizada para mostrar las pantallas que seran utilizadas en el aplicatrivo.

UCFiltroRpt_Cheques



UCGridHistoircoReports



---------- CLASES WINFORM ----------

frmCargarValidacion.vb

este formulario tendrá como funcion cargar la informacion ya validada, serializada que queremos mostrar, incovaraá los metodos como: convertirDAtaClient_Info, Serializar,CompararSchema, TablaMG(Migrar tablas), Insert_TableMG, CargaDataMG_Produccion

frmConection.vb

Este pantalla mostrará el tipo de conexión de la base de datos del clientes con la que vamos a trabajar

```
Imports Cobranza.Configuracion
Imports Cobranza.bussiness
Imports System.Xml
Imports System.IO
Public Class frmConection
   Dim CountWizar As Integer
   Dim tipoConexion As New Estructuras
   Dim ProcesoWizard As New Estructuras
   Dim ManejadorControlWizard As Handler_WizardConeccion
   Dim OconXML As Conexion_Client
   Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnSalir.Click
      Me.Close()
   End Sub
   Private Sub frmConection_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
      IniciarPantalla()
      cbMSSQLProviders.SelectedIndex = 0
      ManejadorControlWizard = New
Handler_WizardConeccion(Estructuras.Conexiones.MsSQL)
   End Sub
   Sub CargaDataConexion()
      Dim oconXml As New Conexion_Client(tipoConexion.TipoConexion)
      Dim oConInfo As List(Of ConexionInfo)
```

```
        oConInfo = oconXml.getInfoConexion()
        Me.lvDetalle.DataSource = oConInfo
    End Sub
    Sub IniciarPantalla()
        gbConfigConeccion.Visible = False
        gbDetalleConneccion.Visible = True
        gbDetalleConneccion.Parent = SplitCMain.Panel1
        gbDetalleConneccion.Dock = DockStyle.Fill
        OcultarGroupBox()
        CargaDataConexion()
    End Sub
    Sub BorrarVinculaciones()
        Dim bRe As New TablasVsXmlBussines
        bRe.DeleteALLRelaciones()
    End Sub
    Sub IniciarConfigCon()
        gbConfigConeccion.Visible = True
        gbConfigConeccion.Parent = SplitCMain.Panel1
        gbConfigConeccion.Dock = DockStyle.Fill
        grbox1.Parent = gbConfigConeccion
        grbox1.Dock = DockStyle.Fill
        grbox1.Visible = True
        CountWizar = 0
    End Sub
    Private Sub btnNuevaCone_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnNuevaCone.Click
        IniciarConfigCon()
    End Sub
    Private Sub btnCerrar_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnCerrar.Click
        IniciarPantalla()
    End Sub
    Private Sub rbMSSQL_CheckedChanged(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles rbMSSQL.CheckedChanged
        ManejadorControlWizard = New
Handler_WizardConeccion(Estructuras.Conexiones.MsSQL)
        tipoConexion.TipoConexion = Estructuras.Conexiones.MsSQL
    End Sub
    Private Sub rbAccess_CheckedChanged(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles rbAccess.CheckedChanged
        ManejadorControlWizard = New
Handler_WizardConeccion(Estructuras.Conexiones.Access)
        tipoConexion.TipoConexion = Estructuras.Conexiones.Access
    End Sub
```

```
    Private Sub btnSiguiente_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnSiguiente.Click
        OcultarGroupBox()
        ProcesoWizard.ProcesoWizard = Estructuras.TipoWizard.Siguiente
        If tipoConexion.TipoConexion = Estructuras.Conexiones.MsSQL Then
            ManejadorControlWizard.HabilitarFrame(ProcesoWizard.ProcesoWizard,
grbox1, grbox2MsSQL, grbox3Ms, grbox1)
        Else
            ManejadorControlWizard.HabilitarFrame(ProcesoWizard.ProcesoWizard,
grbox1, grbox2Access, grbox3Ms, grbox1)
            ManejadorControlWizard.CadenaConexion =
"Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" & BuscarPathAcces()
        End If
        If ManejadorControlWizard.CadenaConexion <> "" Then
            Dim oConXml As New Conexion_Client(tipoConexion.TipoConexion)
            oConXml.setConexionXML(ManejadorControlWizard.CadenaConexion)
            MessageBox.Show("Grabacion Cadena conexion OK")
            BorrarVinculaciones()
            IniciarPantalla()
        End If
    End Sub
    Private Sub btnAtras_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnAtras.Click
        OcultarGroupBox()
        ProcesoWizard.ProcesoWizard = Estructuras.TipoWizard.Atras
        ManejadorControlWizard.HabilitarFrame(ProcesoWizard.ProcesoWizard,
grbox1, grbox2MsSQL, grbox3Ms, grbox1)
    End Sub
    Sub OcultarGroupBox()
        grbox1.Visible = False
        grbox2MsSQL.Visible = False
        grbox3Ms.Visible = False
        grbox2Access.Visible = False
    End Sub
    Function BuscarPathAcces() As String
        Dim myStream As Stream = Nothing
        Dim openFileDialog1 As New OpenFileDialog()
        openFileDialog1.InitialDirectory = "c:\"
        openFileDialog1.Filter = "mdb files (*.mdb)|*..mdb|All files (*.*)|*.*"
        openFileDialog1.FilterIndex = 2
        openFileDialog1.RestoreDirectory = True
        If openFileDialog1.ShowDialog() = System.Windows.Forms.DialogResult.OK
Then
            Try
```

```
            txtPatchAcces.Text = openFileDialog1.FileName
            Return openFileDialog1.FileName
        Catch Ex As Exception
            MessageBox.Show("Cannot read file from disk. Original error: " &
Ex.Message)
            Return ""
        End Try
    End If
    Return ""
  End Function
  Private Sub btnBrowserAcces_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnBrowserAcces.Click
  End Sub
  Private Sub SplitCMain_Panel2_Paint(ByVal sender As System.Object, ByVal e
As System.Windows.Forms.PaintEventArgs) Handles SplitCMain.Panel2.Paint
  End Sub
End Class
```

- **frmContaniner**

  En este formulario se mostrara el menus de los perfiles a usarse en la aplicacion

- **frmLogon.vb**

  Esta pantalla tendra el codigo para que los usuarios con permiso puedan ingresar

- **Frmmantenimiento Reportes.vb**

  Este formulario tendr'a el codigo para modifica, eliminar o crear un nuevo reporte

- **frmMantenimientoUsuario**

  El codigo de este formulario servira para dar mantenimiento al usuario sea este para crear nuevos usuarios, modificarlos o eliminarlos

- **frmViewerReports.vb**

El codigo de este formulario mostrar una lista de reportes a los cuales tendra acceso el usuario dependiendo de la opcion que escoja - 'obtengo la info del usuario logoneado a mi sistema 'para q en base a este se realicen todas las consultas de lo reportes 'en base a los permisos con los q estè cuenta.

- **Vinculador.vb**

Aqui estar'a el codigo para relacionar los campos de cada una de las tablas del cliente con las nuestras.

- **Handler_FilesXML.vb**

Obtines el nodo de cada uno de las tablas relacionadas en el vinculador

CONTROLES INDIVIDUALES USADOS EN EL WINFORM

Para el desarrollo del aplicativo se decidió crear controles ue pedan ahorrar codigo al momento de desarrollar como Los controles indivduales de Combo Boxse que se los creo con la finalidad de poder filtrar la informacion para luego ser reutilizado en otros controles

UCComboCliente.vb

```
Imports Cobranza.bussiness
Public Class UCComboCliente
    Dim oClienteB As New ClienteBussiness(1)
    Sub New()
        ' Llamada necesaria para el Diseñador de Windows Forms.
        InitializeComponent()
        ' Agregue cualquier inicialización después de la llamada aInitializeComponent().
    End Sub
    Sub iniciar()
        'Me.Cmb_Cliente.DataSource = oClienteB.NombreCliente
        Cmb_Cliente.DisplayMember = "nombre"
        Cmb_Cliente.ValueMember = "IdCliente"
    End Sub
```

```
    Private _IdCompania As Integer
    Public Property IdCompania() As Integer
        Get
            Return _IdCompania
        End Get
        Set(ByVal value As Integer)
            _IdCompania = value
        End Set
    End Property
    Private Sub Label1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Label1.Click
    End Sub
    Private Sub Cmb_Cliente_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
Cmb_Cliente.SelectedIndexChanged
    End Sub
End Class
```

UCComboFormaCobro.vb

CONTROLES PARA FILTROS UTILIZADOS EN LOS REPORTES

Estos controles seran utilizados en los reportes con la finalidad de que el usuario pueda visualizar la informacion dependientdo del filtro del ComboBox que este utilizando sea este: UCComboCliente, UCComboEmpres, UCComboSucursal UCComboVendedor, UCComboFormaCobro…

UCFiltroRpt_Cheques.vb

```
Imports Cobranza.bussiness
Imports Cobranza.info
Public Class UCFiltroRpt_Cheques
    Dim CompaniaB As New EmpresaBussiness(1)
    Dim oFormaCobroB As New TipoFormaCobroBussiness(1)
    Dim oSucursalB As New SucursalBussiness(1)
    Dim oClienteB As New ClienteBussiness(1)
    Dim oVendedorB As New VendedorBussiness(1)
    Dim execRpt As New spRpt_ProcesaData_CobrosXClienteBussines 'para ejecutar
reporte
```

```
    Dim idEmpreas As Integer
    Dim idSucu As Integer
    Dim idCobroIni As String
    Dim idCobroFin As String
    Dim idClie As Integer
    Dim idVen As Integer
    Dim fechaIni As Date
    Dim fechaFin As Date
    Dim TipoRpt As String
    Dim NSucursal As String
    Dim NClie As String
    Dim NVend As String
    Dim NTipoRpt As String
    Dim NTipoCobro As String
    Dim NameReports As String
    Public Sub llenaCombos()
        Me.Cmb_Compania.DataSource = CompaniaB.NombreEmpresa
        Me.Cmb_Compania.DisplayMember = "nombre"
        Me.Cmb_Compania.ValueMember = "IdEmpresa"
        Me.Cmb_FormaPago.DataSource = oFormaCobroB.OptenerSoloCheq
        Cmb_FormaPago.DisplayMember = "descripcion"
        Cmb_FormaPago.ValueMember = "IdTipoCobro"
        Me.cmbEstadoChq.DataSource = oFormaCobroB.OptenerEstadoCheq
        cmbEstadoChq.DisplayMember = "descripcion"
        cmbEstadoChq.ValueMember = "IdTipoCobro"
    End Sub
    Private Sub Cmb_Compania_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
Cmb_Compania.SelectedIndexChanged
    End Sub
    Private Sub Cmb_Compania_SelectedValueChanged(ByVal sender As Object,
ByVal e As System.EventArgs) Handles Cmb_Compania.SelectedValueChanged
        Try
            idEmpreas = Me.Cmb_Compania.SelectedValue.ToString
            Me.Cmb_Sucursal.DataSource = oSucursalB.NombreSucrusal(idEmpreas)
            Cmb_Sucursal.DisplayMember = "nombre"
            Cmb_Sucursal.ValueMember = "Codigo"
            idSucu = Cmb_Sucursal.SelectedValue
            Me.Cmb_Cliente.DataSource = oClienteB.NombreCliente(idEmpreas,
idSucu)
            Cmb_Cliente.DisplayMember = "nombre"
            Cmb_Cliente.ValueMember = "IdCliente"
```

Codificacion - Ambiente WEB

CobranzaWeb

Este ambiente utiliza una página web local para subir los reportes y puedan ser visto por los usuarios siempre y cuando tenga permiso

---------- CLASES INFO PARA AMBIETE WEB ----------

D:\Tesis Lourdes\Gestor Cobranza\CobranzaWeb

2.5.1 Codigo para reportes Ambiente Web

Wfrm_BalancexVendedor.aspx

```
Imports Cobranza.info
Imports Cobranza.bussiness
Partial Class Wfrm_BalancexVendedor
    Inherits System.Web.UI.Page
    Dim oParametrosFiltros As New ParametrosInfo

        Get
            setVariables()
            Return oParametrosFiltros
        End Get
    End Property
    Public Sub llenaCombos()
        Me.Cmb_Compania.DataSource = CompaniaB.NombreEmpresa
        Me.Cmb_Compania.DataTextField = "nombre"
        Me.Cmb_Compania.DataValueField = "IdEmpresa"
        Cmb_Compania.DataBind()
        Me.Cmb_FormaPago.DataSource = oFormaCobroB.NombreTipoFormaCobro()
        Cmb_FormaPago.DataTextField = "descripcion"
        Cmb_FormaPago.DataValueField = "IdTipoCobro"
        Cmb_FormaPago.DataBind()
        Dim cia As String = Me.Cmb_Compania.SelectedValue.ToString
        Me.CmbProducto.DataSource = oProductoB.getListaProducto(Val(cia))
        Me.CmbProducto.DataTextField = "descripcion"
        Me.CmbProducto.DataValueField = "IdProducto"
```

```
        CmbProducto.DataBind()
        Me.Cmb_Sucursal.DataSource = oSucursalB.NombreSucrusal(cia)
        Cmb_Sucursal.DataTextField = "nombre"
        Cmb_Sucursal.DataValueField = "Codigo"
        Cmb_Sucursal.DataBind()
        Dim sucu As String = "" 'Me.Cmb_Sucursal.SelectedValue.ToString
        Me.Cmb_Cliente.DataSource = oClienteB.NombreCliente(cia, sucu)
        Cmb_Cliente.DataTextField = "nombre"
        Cmb_Cliente.DataValueField = "IdCliente"
        Cmb_Cliente.DataBind()
        Me.Cmb_Vendedor.DataSource = oVendedorB.NombreVendedor(cia, sucu)
        Cmb_Vendedor.DataTextField = "nombre"
        Cmb_Vendedor.DataValueField = "IdVendedor"
        Cmb_Vendedor.DataBind()
    End Sub
    Sub setVariables()
        oParametrosFiltros.IdCompania = Cmb_Compania.SelectedValue
        oParametrosFiltros.NomEmpresa = Cmb_Compania.SelectedItem.Text
        '-------------------------------------------------------
        oParametrosFiltros.IdSucursalIni = Cmb_Sucursal.SelectedValue
        oParametrosFiltros.IdSucursalFin = IIf(oParametrosFiltros.IdSucursalIni = 0,
500, oParametrosFiltros.IdSucursalIni)
        oParametrosFiltros.NomSucursal = Cmb_Sucursal.SelectedItem.Text
        '-------------------------------------------------------
        oParametrosFiltros.IdClienteIni = Cmb_Cliente.SelectedValue
        oParametrosFiltros.IdClienteFin = IIf(oParametrosFiltros.IdClienteIni = 0, 500,
oParametrosFiltros.IdClienteIni)
        oParametrosFiltros.NomCliente = Cmb_Cliente.SelectedItem.Text
        '-------------------------------------------------------
        oParametrosFiltros.idProductoIni = CmbProducto.SelectedValue
        oParametrosFiltros.idProductoFin = IIf(oParametrosFiltros.idProductoIni = 0,
5000, oParametrosFiltros.idProductoIni)
        oParametrosFiltros.NomProducto = CmbProducto.SelectedItem.Text
        '-------------------------------------------------------
        oParametrosFiltros.idVendedorIni = Cmb_Vendedor.SelectedValue
        oParametrosFiltros.idVendedorFin = IIf(oParametrosFiltros.idVendedorIni = 0,
5000, oParametrosFiltros.idVendedorIni)
        oParametrosFiltros.NomVendedor = Cmb_Vendedor.SelectedItem.Text
        '-------------------------------------------------------
        oParametrosFiltros.IdTipoPagoIni = Cmb_FormaPago.SelectedValue
        If oParametrosFiltros.IdTipoPagoIni.ToLower = "all" Then
            oParametrosFiltros.IdTipoPagoIni = "AAA"
            oParametrosFiltros.IdTipoPagoFin = "ZZZ"
        Else
```

```
            oParametrosFiltros.IdTipoPagoFin = oParametrosFiltros.IdTipoPagoIni
        End If
        oParametrosFiltros.NomTipoPago = Cmb_FormaPago.SelectedItem.Text
    End Sub
    Protected Sub Cmb_Sucursal_SelectedIndexChanged(ByVal sender As Object,
ByVal e As System.EventArgs) Handles Cmb_Sucursal.SelectedIndexChanged
        setVariables()
    End Sub
    Protected Sub Cmb_Vendedor_SelectedIndexChanged(ByVal sender As Object,
ByVal e As System.EventArgs) Handles Cmb_Vendedor.SelectedIndexChanged
        setVariables()
    End Sub
    Protected Sub CmbProducto_SelectedIndexChanged(ByVal sender As Object,
ByVal e As System.EventArgs) Handles CmbProducto.SelectedIndexChanged
        setVariables()
    End Sub
    Protected Sub Cmb_Compania_SelectedIndexChanged(ByVal sender As Object,
ByVal e As System.EventArgs) Handles Cmb_Compania.SelectedIndexChanged
        setVariables()
    End Sub
    Protected Sub Cmb_FormaPago_SelectedIndexChanged(ByVal sender As Object,
ByVal e As System.EventArgs) Handles Cmb_FormaPago.SelectedIndexChanged
        setVariables()
    End Sub
    Protected Sub Cmb_Cliente_SelectedIndexChanged(ByVal sender As Object,
ByVal e As System.EventArgs) Handles Cmb_Cliente.SelectedIndexChanged
        setVariables()
    End Sub
    Protected Sub Page_Load(ByVal sender As Object, ByVal e As
System.EventArgs) Handles Me.Load
        If Not IsPostBack Then
            llenaCombos()
        End If
    End Sub
End Class
```

Codificación: Ambiente - Móvil

CobranzaSmartDevice

AMBIENTE MOVIL.: Estamos utilizando el Emulador de Visual 2008 para poder subir los reportes en el móvil, y el web service que es la página en la cual publicaremos todos los métodos para poderlos ver desde cualquier máquina para ello utiliza el programa Microsoft ActiveSync, mostrando la interoperabilidad de la aplicación.

Fem_Index.vb

```
Public Class frm_Index
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
    System.EventArgs) Handles Button1.Click
     frm_RecaudacionFilter.Show()
    End Sub
End Class
```

D:\Tesis Lourdes\Gestor Cobranza\CobranzaSmarkDevice

Fem_RecaudacionRPT

```
Imports System.Net
Public Class frm_RecaudacionRPT
Private Sub frm_RecaudacionRPT_Load(ByVal sender As Object, ByVal e As
System.EventArgs) Handles MyBase.Load
Me.WebBroRpt.Navigate(New
Uri("http://lacivarw2003s/ReportServer/Pages/ReportViewer.aspx?%2fCobranzaRep
orts%2fRpt_Recaudaciones&rs:Command=Render")) End Sub End
```

Frm_RecaudacionFilter.vb

```
Public Class frm_RecaudacionFilter
    Dim parametros As New ParametrosInfo

    Private Sub frm_RecaudacionFilter_KeyDown(ByVal sender As
System.Object, ByVal e As System.Windows.Forms.KeyEventArgs) Handles
MyBase.KeyDown
        If (e.KeyCode = System.Windows.Forms.Keys.Up) Then
            'Up
        End If
        If (e.KeyCode = System.Windows.Forms.Keys.Down) Then
            'Down
        End If
        If (e.KeyCode = System.Windows.Forms.Keys.Left) Then
            'Left
        End If
        If (e.KeyCode = System.Windows.Forms.Keys.Right) Then
            'Right
        End If
        If (e.KeyCode = System.Windows.Forms.Keys.Enter) Then
            'Enter
        End If
    End Sub

    Sub setParametros()
        parametros.IdCompania = cmb_Empresa.SelectedValue
        parametros.IdSucursalIni = cmb_sucursal.SelectedValue
        parametros.IdSucursalFin = cmb_sucursal.SelectedValue
        If parametros.IdSucursalIni = 0 Then
            parametros.IdSucursalIni = 1
            parametros.IdSucursalFin = 100
        End If
        parametros.FechaIni = dtp_fechaIni.Value
        parametros.FechaFin = dtp_fechaFin.Value
        parametros.IdClienteIni = cmb_cliente.SelectedValue
        parametros.IdClienteFin = cmb_cliente.SelectedValue
        If parametros.IdClienteIni = 0 Then
            parametros.IdClienteIni = 0
            parametros.IdClienteFin = 99999
        End If
        parametros.idProductoIni = cmb_producto.SelectedValue
        parametros.idProductoFin = cmb_producto.SelectedValue
        If parametros.idProductoIni = 0 Then
            parametros.idProductoIni = 1
            parametros.idProductoFin = 99999
        End If
        parametros.IdTipoPagoIni = cmb_tipoPago.SelectedValue
        parametros.IdTipoPagoFin = cmb_tipoPago.SelectedValue
        If parametros.IdTipoPagoIni = "All" Then
            parametros.IdTipoPagoIni = "AAA"
            parametros.IdTipoPagoFin = "ZZZ"
        End If
        parametros.idVendedorIni = cmb_vendedor.SelectedValue
        parametros.idVendedorFin = cmb_vendedor.SelectedValue
```

```
        If parametros.idVendedorIni = 0 Then
            parametros.idVendedorIni = 1
            parametros.idVendedorFin = 5000
        End If
        parametros.NomCliente = cmb_cliente.Text
        parametros.NomEmpresa = cmb_Empresa.Text
        parametros.NomProducto = cmb_producto.Text
        parametros.NomSucursal = cmb_sucursal.Text
        parametros.NomTipoPago = cmb_tipoPago.Text
        parametros.NomVendedor = cmb_vendedor.Text
        parametros.NumFacIni = 0
        parametros.NumFacFin = 9999999
        parametros.sTipoRpt = "Consulta_ACT"
    End Sub

    Private Sub frm_RecaudacionFilter_Load(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
        Dim wsEmp As New WSCobranza.WSCobranza
        Dim clase As New List(Of WSCobranza.SucursalInfo)

        Me.cmb_Empresa.DataSource = wsEmp.GetEmpresas
        Me.cmb_Empresa.ValueMember = "IdEmpresa"
        Me.cmb_Empresa.DisplayMember = "nombre"
        Me.cmb_sucursal.DataSource = wsEmp.GetSucursales(1)
        Me.cmb_sucursal.ValueMember = "IdSucursal"
        Me.cmb_sucursal.DisplayMember = "nombre"
        Me.cmb_vendedor.DataSource = wsEmp.GetVendedor(1, 1)
        Me.cmb_vendedor.ValueMember = "IdVendedor"
        Me.cmb_vendedor.DisplayMember = "nombre"
        Me.cmb_cliente.DataSource = wsEmp.GetClientes(1, 1)
        Me.cmb_cliente.DisplayMember = "nombre"
        Me.cmb_cliente.ValueMember = "IdCliente"
        Me.cmb_producto.DataSource = wsEmp.GetProductos(1)
        Me.cmb_producto.DisplayMember = "descripcion"
        Me.cmb_producto.ValueMember = "IdProducto"
        Me.cmb_tipoPago.DataSource = wsEmp.GetFormaPago
        Me.cmb_tipoPago.DisplayMember = "descripcion"
        Me.cmb_tipoPago.ValueMember = "IdTipoCobro"
    End Sub

    Private Sub btn_consultar_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btn_consultar.Click
        Dim execRpt As New WSCobranza.WSCobranza
        setParametros()
        execRpt.Ejecutar_SP_Reports_Recaudacion()
           parametros.IdCompania, parametros.IdSucursalIni,
parametros.IdTipoPagoIni, _
           parametros.IdTipoPagoFin, parametros.IdClienteIni,
parametros.idVendedorIni, _
           parametros.FechaIni, parametros.FechaFin, "A", "Z",
parametros.sTipoRpt, _
           parametros.NomSucursal, parametros.NomCliente,
parametros.NomVendedor, _
           "", parametros.NomTipoPago, "Recaudacion ", 0)
```

```
        frm_RecaudacionRPT.Show()
```